UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

Lucas Tavares Zanuzzo

Sistema de monitoramento do consumo de água controlado por um aplicativo Android

São Carlos

2017

Lucas Tavares Zanuzzo

Sistema de monitoramento do consumo de água controlado por um aplicativo Android

Dissertação apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo, para obtenção do título de Engenheiro de Computação.

Orientador: Prof. Dr. Maximiliam Luppe

São Carlos

2017

Tavares-Zanuzzo, Lucas
T27s Sistema de monitoramento do consumo de água controlado por um aplicativo Android / Lucas Tavares-Zanuzzo; orientador Maximiliam Luppe. São Carlos, 2017.

Monografia (Graduação em Engenharia de Computação) -- Escola de Engenharia de São Carlos e Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo, 2017.

1. Sistemas Embarcados. 2. Android. 3. Bluetooth. 4. Bancos de Dados. I. Título.

FOLHA DE APROVAÇÃO

Nome: Lucas Tavares Zanuzzo

Título: "Sistema de monitoramento do consumo de água controlado por um aplicativo Android"

Trabalho de Conclusão de Curso defendido em 27/06/2017.

Comissão Julgadora:	**Resultado:**
Prof. Dr. Maximiliam Luppe (Orientador) - SEL/EESC/USP	Aprovado
Prof. Assistente Edson Gesualdo SEL/EESC/USP	Aprovado
Prof. Associado Evandro Luis Linhari Rodrigues SEL/EESC/USP	Aprovado

Coordenador do Curso Interunidades Engenharia de Computação pela EESC:

Prof. Dr. Maximiliam Luppe

Este trabalho é dedicado ao meu avô, que deixou muitas saudades.
E a Rafaella, que acabou de chegar a esse mundo.

AGRADECIMENTOS

À Universidade de São Paulo, seus professores e funcionários que foram responsáveis pela minha formação acadêmica, pessoal e profissional.

Aos meus pais que possibilitaram que tudo isso seja possível, que me deram sempre o apoio para me tornar uma pessoa melhor e seguir meus sonhos.

Aos meus amigos da faculdade Zuera, Boka, Pocs, Robs, Nub, Zé, Nóia, Lui, Falqueto, Dilce, Lino entre outros que participaram da minha vida acadêmica.

À Ligia, Laura e Isadora que compartilharam comigo o melhor ano da minha vida, e estiveram ao meu lado em um momento de grande transformação pessoal.

À Carol pelo apoio na realização desse trabalho.

Aos meus amigos de infância Neto, Taiane, Luiz, Felipe, Marianinha, Paulo, Murilo e Malu que sempre me deram muito apoio e nunca deixaram a distância abalar nossa amizade.

"Não existe mal maior que a raiva, nem virtude maior que a paciência."
Geshe Kelsang Gyatso

RESUMO

TAVARES ZANUZZO, L. **Sistema de monitoramento do consumo de água controlado por um aplicativo Android**. 2017. 63p. Dissertação - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2017.

A economia do consumo de água engloba uma série de demandas atuais, tanto na esfera ecológica quanto na financeira. Eventos recentes como a grande crise hídrica no estado de São Paulo (2014-2016) mostram a importância da conscientização desse consumo. O consumidor brasileiro conhece pouco sobre seu consumo, conhecimento em sua maioria, composto apenas pelo valor do consumo mensal enviado pelas secretarias de água e esgoto. Um acompanhamento mais detalhado, poderia desencadear uma cultura de consumo consciente e consequentemente uma economia financeira. Este trabalho implementa um sistema de monitoramento do consumo de água controlado através de uma aplicação Android. O sistema permite que o usuário monitore a vazão de um líquido que atravessa um determinado cano ou mangueira, utilizando um medidor de fluxo. O medidor pode ser anexado em diferentes pontos pelo usuário, como hidrômetros, máquinas de lavar roupas, torneiras e irrigadores. O medidor é conectado a uma plataforma embarcada Raspberry Pi 3 Model B, que armazena os dados em uma banco de dados SQLite e comunica-se com o aplicativo utilizando Bluetooth. Utilizando o aplicativo, o usuário pode acompanhar a leitura do medidor, determinar o inicio e final das leituras e a verificação das medidas feitas de forma gráfica e no formato de histórico. Também é possível determinar parâmetros para as leituras, sendo eles, uma tarifa média para avaliar os custos, uma meta de consumo e um valor mínimo para cobrança da tarifa. O sistema foi testado em um ambiente residencial, registrando o consumo de uma torneira por determinado tempo e a lavagem de uma máquina de lavar roupas.

Palavras-chave: Sistemas Embarcados. Android. Bluetooth. Bancos de Dados.

LISTA DE ILUSTRAÇÕES

Figura 1 – Tipico medidor de vazão por obstrução. 23
Figura 2 – Tipico medidor de vazão por turbina. 24
Figura 3 – Esquemático de um medidor de vazão eletromagnético. 25
Figura 4 – Esquemático de medidor de vazão por efeito Doppler. 26
Figura 5 – Esquemático de medidor de vazão por tempo de passagem. 27
Figura 6 – Sensor de efeito Hall 27
Figura 7 – Sensor de efeito Hall aplicado um campo magnético 28
Figura 8 – Interior do medidor de vazão 28
Figura 9 – Visão geral do projeto 33
Figura 10 – Medidor de vazão YF-S201 34
Figura 11 – Raspberry Pi 3 Model B 36
Figura 12 – RTC DS3231 36
Figura 13 – Estrutura do banco de dados utilizado no projeto 37
Figura 14 – Diagrama de fluxo do *script* de leitura do sensor. 39
Figura 15 – Diagrama de fluxo do *script* de comunicação com o aplicativo. 41
Figura 16 – Menu lateral do aplicativo 43
Figura 17 – Classe Principal do aplicativo 44
Figura 18 – Classe Histórico, tela inicial 45
Figura 19 – Fluxograma do algoritmo que recebe a stream de dados 46
Figura 20 – Classe Histórico, gráfico apresentado 47
Figura 21 – Classe Meta e Tarifa 48
Figura 22 – Classe Raspberry do aplicativo 49
Figura 23 – Instalação do medidor de vazão 51
Figura 24 – Sistema de testes 51
Figura 25 – Teste com o medidor calibrado 52
Figura 26 – Teste com o medidor descalibrado 53
Figura 27 – Teste com o máquina de lavar 53

LISTA DE TABELAS

Tabela 1 – Resultados da calibração do medidor de vazão. 35

LISTA DE ABREVIATURAS E SIGLAS

UHF	*Ultra High Frequency*
RF	Rádio Frequência
RFCOMM	*RadioFrequency Communications*
API	*Application Programming Interface*
UUID	*Universally Unique Identifiers*
ID	*Identity*
ABS	*Anti-lock Breaking System*
PC	*Personal Computer*
DC	*Direct current*
PWD	*Pulse-Width Modulation*
RTC	*Real Time Clock*
GPIO	*General Pin Input Output*
I^2C	*Inter Integrated Circuit*
CPU	*Central Processing Unit*
SQL	*Structured Query Language*
SGBD	Sistema Gerenciador de Bancos de Dados
ACID	*Atomicity, Consistency, Isolation, Durability*
UTC	*Universal Time Coordinated*
ASCII	*American Standard Code for Information Interchange*
NTS	Norma Técnica Sabesp.

SUMÁRIO

1 INTRODUÇÃO 21
1.1 Motivação 21
1.2 Objetivos 21
1.3 Estrutura deste documento 22

2 EMBASAMENTO TEÓRICO 23
2.1 Métodos de medida de vazão 23
2.1.1 Medidores por obstrução 23
2.1.2 Medidores por turbinas 24
2.1.3 Medidores eletromagnéticos 25
2.1.4 Medidores por ultrassom 26
2.2 Sensor de efeito Hall 27
2.3 Bluetooth 29
2.4 Sistemas embarcados 30
2.5 Considerações finais 31

3 DESENVOLVIMENTO DO TRABALHO 33
3.1 Projeto 33
3.2 Medidor de vazão e calibração 34
3.3 Sistema embarcado 36
3.4 Banco de dados 37
3.5 Script de leitura do sensor 38
3.6 Script de comunicação com o aplicativo 38
3.7 Aplicativo Android 41
3.7.1 Classe Principal 42
3.7.2 Classe Histórico 44
3.7.3 Classe Meta e Tarifa 47
3.7.4 Classe Raspberry 48
3.7.5 Classe Gráfico 48
3.8 Considerações finais 49

4 TESTES, RESULTADOS E DISCUSSÕES 51
4.1 Considerações finais 53

5 CONCLUSÃO 55
5.1 Trabalhos futuros 55

REFERÊNCIAS 57

ANEXOS 59

ANEXO A – ESPECIFICAÇÕES TÉCNICAS DO MEDIDOR DE FLUXO YF-S201 61

ANEXO B – ESPECIFICAÇÕES TÉCNICAS DA LAVADORA CONSUL CWI07A 63

1 INTRODUÇÃO

1.1 Motivação

A água é um recurso natural presente em várias partes do nosso dia a dia e quanto maior o consumo, maior o preço, qualquer descuido ou desperdício poderá custar caro. Segundo uma pesquisa do Ibope, em (DURAO, 2012) uma grande quantidade de consumidores no Brasil admite ter pouco controle sobre o seu consumo de água. A otimização do consumo de água é importante tanto na área financeira quanto na ambiental, desenvolvendo um futuro mais sustentável.

Na parte residencial, o consumo de água é medido mensalmente pelas secretarias de água e esgoto e informam apenas a quantidade de água despendida durante o mês, uma informação muito superficial e pouco explicativa. Não é possível determinar a distribuição do consumo, o consumo parcial até o momento ou o consumo de um aparelho em específico, por exemplo, uma máquina de lavar roupas ou irrigadores.

Eventos recentes, como a grande crise hídrica no estado de São Paulo nos anos de 2014 à 2016 (ROSSI, 2014)(COHEN, 2015), onde os níveis de seca e redução na oferta de água atingiram níveis muito preocupantes, mostram a importância da economia de recursos hídricos no cenário atual. Uma ferramenta para a análise do consumo seria de grande ajuda para a conscientização da sociedade sobre o consumo de água, para economia de gastos e também para melhorar a percepção do cidadão sobre seu consumo de água residencial.

1.2 Objetivos

O objetivo do presente trabalho é projetar um sistema de monitoramento do consumo de água que passe por um determinado cano ou mangueira. Um sistema embarcado com um medidor de vazão, que mede a vazão e quantidade de líquido, realiza as leituras de consumo. Essas leituras serão armazenadas em um banco de dados interno e um aplicativo móvel atuará como interface gráfica para o sistema.

O sistema permitirá o usuário ter o controle sobre a leitura do sensor, ou seja, ele poderá realizar as leituras no intervalo de tempo desejado e consultar um histórico com todas as leituras feitas. A partir do histórico, será possível realizar uma leitura gráfica, para verificar a distribuição do consumo de água no intervalo de tempo determinado. Também será possível excluir leituras e definir parâmetros de tarifa e meta para controlar os gastos. Quando uma leitura estiver sendo realizada, será possível acompanhar a leitura no aplicativo, exibindo a quantidade parcial de água consumida e a vazão atual entre outras informações.

O sistema será *plug-and-play*, sem necessidade de qualquer configuração inicial no *hardware*, facilitando o uso da solução sem maiores configurações técnicas. O sistema embarcado escolhido foi a Raspberry pi 3 model B, que é uma plataforma embarcada de propósito geral, altamente difundida, de fácil acesso e baixo custo. O aplicativo móvel desenvolvido para o projeto será implementado em um sistema operacional Android.

O Android é um sistema operacional preferencialmente desenvolvido para dispositivos móveis, desenvolvido atualmente pela Google e baseado em um *kernel* Linux. Os *smartphones* estão amplamente difundidos entre a população brasileira chegando a 170 milhões de aparelhos (FOLHA DE SãO PAULO, 2016). Segundo uma pesquisa realizada pela Kantar (KANTAR, 2017a), dos aparelhos vendidos no brasil em janeiro de 2017, 93% tinham o Android como sistema operacional (KANTAR, 2017b).

Logo, esse tipo de aplicação seria de fácil acesso e de utilização por grande parte dos usuários brasileiros, evitando assim, a necessidade de aquisição de telas de qualquer tipo para a visualização dos dados. Toda a comunicação entre o sistema embarcado e o *smartphone* é feita por Bluetooth, o qual é um padrão de comunicação sem fio para curtas distâncias. Usando Bluetooth, não é necessário uma conexão com a *internet*, assim, facilita o uso do sistema em áreas afastadas da rede, como por exemplo plantações.

1.3 Estrutura deste documento

O trabalho começa com uma introdução, apresentando os objetivos e motivações que foram a base para o desenvolvimento desse projeto e também uma breve descrição das funcionalidades do sistema proposto. Em seguida, será apresentada uma fundamentação teórica fundamental para o entendimento do restante do conteúdo do trabalho. Posteriormente, o desenvolvimento do projeto será apresentado, ilustrando a implementação, decisões, materiais e métodos utilizados. Por fim, serão apresentados os resultados obtidos nos testes realizados, seguido de uma conclusão sobre seus resultados.

2 EMBASAMENTO TEÓRICO

2.1 Métodos de medida de vazão

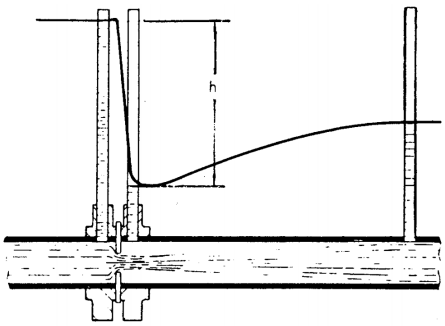
Para medir a vazão de um fluido em determinado meio, existem diferentes dispositivos, técnicas e princípios físicos que podem ser utilizados. Como o foco desse trabalho é a medição da vazão de líquidos, o conteúdo dessa seção será dedicado aos dispositivos mais utilizados para esse objetivo. O conteúdo da seção tem como base a referência em (UPP, 2002).

2.1.1 Medidores por obstrução

Os medidores de vazão baseados em obstrução, são os mais comuns, utilizados e também os mais antigos. Métodos muito antigos, como o proposto por Venturi em 1797, ainda são utilizados atualmente em suas versões mais modernas. Os medidores mais comuns são do tipo placa de orifício, bocal e ainda Venturi. Seu funcionamento se baseia em determinar a vazão de determinado líquido, à partir da variação de pressão entre dois pontos no escoamento.

Nesse tipo de medidor o dispositivo principal, um restritor, consiste de um elemento que restringe o fluxo de líquido no tubo como ilustrado na Figura 1. Essa restrição causa uma diferença de pressão (queda de pressão) através do restritor e essa diferença na pressão pode ser relacionada com a vazão do líquido através da equação de Bernoulli. A equação de Bernoulli apenas é válida quando a área física para o escoamento do líquido se mantém após atravessar o restritor.

Figura 1: Tipico medidor de vazão por obstrução.



Fonte: Retirada de (UPP, 2002)

Porém, como o tubo utilizado mantém seu diâmetro após o restritor, alguns ajustes devem ser feitos para adaptar a equação a diferença de áreas. Esse ajuste, chamado de

"Coeficiente de Descarga"relaciona o diâmetro do tubo com a pressão diferencial no local do restritor e o número de Reynolds.

Os coeficientes de descarga são conhecidos empiricamente e tabelados em função do número de Reynolds e do diâmetro do tubo. Note que, esses dados tabelados são utilizados para a construção dos restritores, caso ele possua uma relação não tabelada, novas medidas empíricas devem ser feitas para a calibração do medidor. Logo, sua fabricação normalmente segue as relações presentes nessas tabelas.

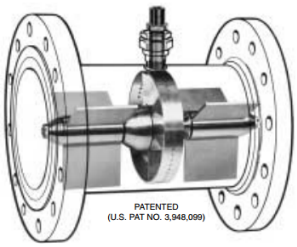
O outro componente presente no medidor, é a unidade de medição da pressão diferencial no tubo. Saber a pressão diferencial é necessário para determinar a vazão a partir da equação de Bernoulli adaptada. Na Figura 1, a medição da pressão diferencial é feita a partir de uma tubulação vertical. Também é necessário conhecer a temperatura e composição do líquido, variáveis presentes na equação de Bernoulli.

Esses medidores possuem a vantagem de serem muito difundidos e documentados, possuem fabricação simples, de baixo custo, fácil manutenção e não requer energia elétrica para seu funcionamento. Isso faz com que eles sejam amplamente utilizados, porém normalmente não são utilizados nos casos de líquidos viscosos, lamas e líquidos não-Newtonianos.

2.1.2 Medidores por turbinas

O medidor de vazão utilizando turbinas é o mais utilizado atualmente, atuando por exemplo, nos hidrômetros residenciais. Um exemplo é ilustrado na Figura 2. Sua teoria de funcionamento simples e intuitiva contrasta com uma construção complexa, levando em consideração fatores como angulação e número de lâminas, mancais de fixação e a qualidade do rolamento.

Figura 2: Tipico medidor de vazão por turbina.



Fonte: Retirada de (UPP, 2002)

O medidor basicamente é um medidor de velocidade, onde um rotor de giro livre é colocado contra o escoamento do líquido. Ele é posicionado de forma que, o centro do rotor seja coaxial com o centro do duto, e que o rotor gire quando houver a presença de um escoamento. O arrasto do escoamento nas pás da turbina gera uma velocidade angular

no rotor, de modo que essa velocidade seja proporcional a vazão do líquido. Algum sensor eletrônico deve ser utilizado para medir a velocidade angular do rotor, para a leitura e determinação da vazão.

A tubulação deve ser preparada para que, no escoamento, sejam eliminados perfis de velocidade não padronizados e turbulências. Além disso, na calibração deve-se levar em conta o atrito do rolamento e também a pequena quantidade de líquido que passa antes de romper o atrito estático do rotor.

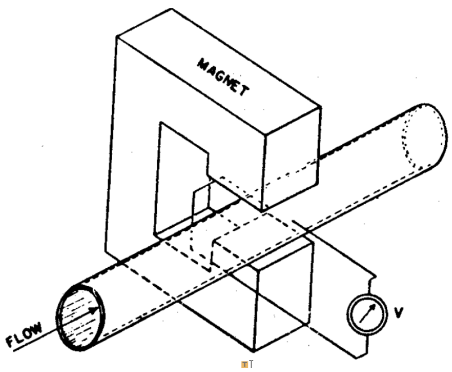
Os medidores de turbina possuem versões de alto e baixo custo, dependendo da precisão necessária na aplicação. No geral, possuem uma boa precisão, devido a vantagem de usar um sensor de velocidade eletrônico que pode transmitir em altas taxas. A manutenção deve ser periódica, fazendo uma inspeção no funcionamento e integridade do rotor, além da limpeza de detritos que, eventualmente, ficam presos nas pás. Como o rolamento tem um desgaste, uma recalibração também é necessária após um certo período de operação.

2.1.3 Medidores eletromagnéticos

Os medidores de vazão eletromagnéticos são utilizados para medir a vazão de líquidos condutores. Como o medidor não é intrusivo no encanamento, a densidade e viscosidade do líquido não interferem na aferição, fazendo com que ele seja muito utilizado para calcular a vazão de líquidos lamacentos ou corrosivos.

Inicialmente temos um campo magnético aplicado no encanamento, como ilustrado na Figura 3. A vazão do líquido provoca uma alteração na densidade de fluxo eletromagnético criada na seção do tubo. Para um escoamento com velocidade média, uma diferença de potencial é observada entre os dois eletrodos transversais, como ilustrado na Figura 3. Esse é um resultado direto da Lei da indução de Faraday, em condições ideais. Como a vazão pode ser calculada a partir dessa velocidade e a área do tubo, temos que a vazão pode ser determinada a partir da diferença de potencial elétrico entre dois polos transversais ao campo magnético.

Figura 3: Esquemático de um medidor de vazão eletromagnético.



Fonte: Retirada de (UPP, 2002)

Esse medidor tem a vantagem de poder operar em ambas as direções de escoamento sem necessitar qualquer adaptação. Também tem a vantagem de não ter contato com as tubulações e não possui peças móveis que desgastam rapidamente, fazendo com que a manutenção seja mais fácil e não tenha problemas com acumulo de dejetos. Entretanto, esse tipo de medidor possui um alto custo de fabricação, consumo de energia elétrica, peso e tamanho elevados.

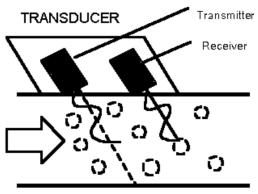
2.1.4 Medidores por ultrassom

Seu funcionamento está baseado nas características de propagação do som em um meio, mais precisamente, como o sinal sonoro é modificado ou refletido dependendo da intensidade da vazão presente na tubulação. Existem dois principais tipos de medidores ultrassônicos, aqueles baseados em efeito Doppler e outro por tempo de passagem.

O efeito Doppler é caracterizado pelas alterações na frequência de uma onda, causadas por um movimento relativo entre a fonte emissora e um receptor. O medidor por efeito Doppler, ilustrado na Figura 4, possui dois sensores ultrassônicos dispostos lado a lado, o mais a esquerda é um emissor e o mais direita um receptor. O emissor envia ondas sonoras que colidem com partículas presente no líquido. Essa onda é refletida pela partícula e é capturada pelo receptor. A partícula que refletirá a onda sonora está se movimentando no fluido com a mesma velocidade do líquido, e o receptor conseguirá determinar, a partir das alterações na frequência de onda a velocidade da partícula, devido ao Efeito Doppler. Como a velocidade da partícula é a mesma do líquido, é possível determinar a vazão do encanamento.

Note que, para o funcionamento desse sensor, o líquido utilizado deve conter partículas suspensas ou bolhas para a reflexão do sinal. Essas partículas devem possuir uma quantidade e distribuição suficientes para o calculo da vazão e também um tamanho considerado para atingir boas reflexões.

Figura 4: Esquemático de medidor de vazão por efeito Doppler.

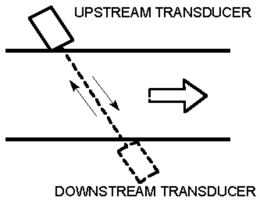


Fonte: Retirada de (UPP, 2002)

Um medidor de tempo de passagem é ilustrado na Figura 5. Dois sensores ultrassônicos são dispostos em lados opostos no encanamento, e então calcula-se a diferença entre os tempos de propagação dos sinais sônicos através do duto entre dos dois sensores. Os tempos de propagação da onda, a favor e contra o fluxo, são diferentes, porque dependem

da direção da velocidade do fluido em que ele se propaga. A partir da diferença dos tempos é possível determinar a velocidade do fluido. Esse medidor tem a vantagem de poder operar nos dois sentidos de escoamento, pois muda-se apenas o referencial entre os sensores ultrassônicos. Porém, a presença de bolhas ou detritos nesse medidor, podem afetar a intensidade da onda sonora e a qualidade da medida.

Figura 5: Esquemático de medidor de vazão por tempo de passagem.



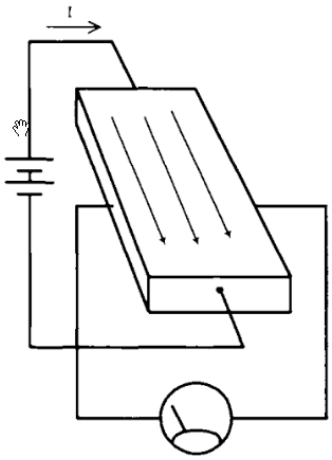
Fonte: Retirada de (UPP, 2002)

Ambos os medidores por ultrassom, não possuem partes móveis e não estão em contato direto com o escoamento, diminuindo o seu desgaste e facilitando sua manutenção. Sensores ultrassônicos de alta qualidade podem tornar esse tipo de medidor muito preciso, porém com um aumento considerável no custo juntamente com o consumo de energia elétrica.

2.2 Sensor de efeito Hall

O sensor de efeito Hall é um transdutor, ou seja, é um dispositivo capaz de transformar um tipo de energia em outra. Com base na referência (EDWARD, 2006) será explicado seu funcionamento.

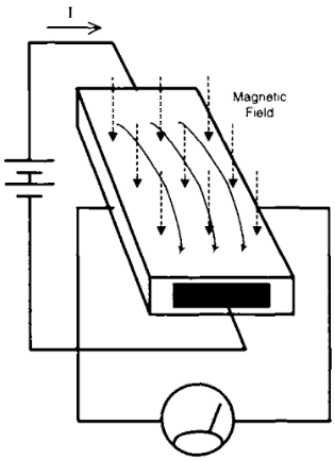
Figura 6: Sensor de efeito Hall



Fonte: Retirada de (EDWARD, 2006)

A Figura 6 ilustra uma fina lâmina de material condutivo, como o cobre, por onde passa uma corrente elétrica causada por uma diferença de potencial na lâmina, neste caso usando uma bateria. Um multímetro é colocado para medir a tensão na direção oposta a conexão da bateria, nessa situação, como ilustrado na Figura 6, o multímetro mede zero volts. Quando um campo magnético é aplicado na lâmina com o ângulo correto em relação ao fluxo da corrente, como na Figura 7, uma pequena diferença de potencial é observada através da lâmina pelo multímetro. Caso a direção (polaridade) do campo magnético seja revertida, o mesmo acontecerá com a tensão induzida pelo campo na placa. Esse fenômeno é chamado de efeito Hall.

Figura 7: Sensor de efeito Hall aplicado um campo magnético



Fonte: Retirada de (EDWARD, 2006)

No presente trabalho, utilizou-se um medidor de vazão por turbina que possui um sensor de efeito Hall magnético em seu interior, a fim de determinar a quantidade de água e a vazão que atravessa um determinado cano ou torneira. O medidor contém em seu interior uma pequena hélice, em detalhe na Figura 8. A hélice é colocada na direção oposta à vazão, para se mover com a passagem do líquido, fazendo com que a velocidade de rotação da hélice seja proporcional a velocidade de escoamento do líquido.

Figura 8: Interior do medidor de vazão

Fonte: Imagem produzida pelo autor

Um sensor de efeito Hall magnético é colocado próximo da região de rotação das pás, e um ímã é colocado na extremidade de uma das pás. No medidor utilizado no projeto, o campo magnético é induzido por um ímã presente na extremidade de uma das pás da turbina. O metal no interior do sensor é energizado por um par de fios e a tensão de saída é enviado por um terceiro fio. Assim, a cada revolução completa da turbina é emitido um pulso elétrico pelo sensor. Contando o número de pulsos elétricos emitidos devido a passagem de uma certa quantidade de líquido, é possível calcular facilmente a quantidade de água que atravessa o sensor e, calculando a frequência de pulsos, é possível calcular a vazão.

2.3 Bluetooth

O Bluetooth é uma tecnologia de comunicação sem fio entre dois dispositivos para curtas distâncias usando ondas de rádio UHF. Para o Bluetooth, existem muitas implementações diferentes, porém algumas características básicas permanecem as mesmas. Para dois dispositivos se conectarem, ambos devem ter o *hardware* e o *software* Bluetooth. O *hardware*, independente da implementação, apresenta um *transceiver* RF de banda base e os *softwares* apresentam protocolos que possibilitam os dispositivos se conectarem e trocarem dados entre si.

O tema Bluetooth engloba uma gama muito grande de assuntos diferentes, como frequências de transmissão, modulação e demodulação de sinais e protocolos para transmissão de áudio. Para a revisão bibliográfica será usado como base a referência (ALBERT, 2007) o qual destaca com clareza como acontece a conexão, ponto mais importante desse projeto. Superficialmente temos os seguintes passos:

- Encontrar o dispositivo que deseja-se conectar.
- Determinar o protocolo para a conexão.
- Efetuar uma conexão de entrada e/ou saída de dados.
- Enviar e receber dados.

Inicialmente há o processo de procura e detecção dos dispositivos, cada dispositivo possui um ID no chip de 48 bits único para identificação e também tem um nome amigável para o usuário o qual é o nome do dispositivo.

Após o dispositivo determinar em qual ele deseja se conectar, inicia-se a parte de escolha do protocolo de comunicação. O usuário irá utilizar o protocolo que melhor se adapta a sua aplicação entre os disponíveis. No projeto será utilizado o RFCOMM, pois ambos os dispositivos utilizados no projeto suportam a implementação. O RFCOMM transporta os dados a partir de *streams*, que simula o comportamento de uma comunicação

serial RS-232. Essa tecnologia é confiável, simples de utilizar e altamente difundida. Grande parte dos sistemas operacionais dão suporte a RFCOMM, apresentando várias API's. Aplicações que usam portas seriais podem ser rapidamente portadas para usar RFCOMM.

Uma vez que o endereço numérico e o protocolo de transporte são conhecidos, é necessário definir qual porta será utilizada para a comunicação e qual o serviço que será utilizado. O UUID é um número de 128 bits, que identifica qual serviço será utilizado. Esse identificador é um padrão no desenvolvimento de softwares (THE INTERNET SOCIETY, 2005) e no projeto, por exemplo, é utilizado o *Standard SerialPortService ID*, utilizado para serviços de comunicação serial.

Definida a porta e o tipo de serviço, o restante da comunicação ocorre por *sockets*, que representam o final da conexão. Sua utilização é padrão em qualquer configuração de rede. Quando criado pela primeira vez, o *socket* ainda não está conectado e não pode ser usado ainda para comunicação. Para conectá-lo, no entanto, o aplicativo deve decidir se ele será usado como um *socket* de servidor para ouvir conexões de entrada, ou como um *socket* de cliente para estabelecer uma conexão de saída.

Caso seja definido como um *socket* de cliente, o programa cliente irá chamar um comando de conexão, especificando qual dispositivo irá se conectar, e em qual porta. O sistema operacional então cuida de todos os detalhes de nível inferior, reservando recursos no adaptador Bluetooth local, procurando o dispositivo remoto e estabelecendo uma conexão. Uma vez conectado o socket, ele pode ser usado para transferência de dados.

Caso seja um *socket* de servidor, existem três passos que a aplicação deve seguir. Primeiro o *socket* deve ser vinculado aos recursos locais do Bluetooth, enviando a porta e o endereço do adaptador. Segundo, ele deve ser configurado para o modo de escuta (*listening*). Isso indica ao sistema operacional que deve-se escutar a porta e o adaptador que foram vinculados ao *socket*. E terceiro, a aplicação utiliza o socket para receber as conexões.

2.4 Sistemas embarcados

Atualmente existem muitas definições diferentes para um sistema embarcado, e a partir da referência (TAMMY, 2005) pode-se notar algumas características comuns entre essas definições. Para o autor, um sistema embarcado é um sistema computacional que possui uma função mais específica e não um computador de propósito geral como um PC. Porém, o autor admite que ainda não há um consenso coletivo sobre a definição. Em geral um sistema embarcado possui as seguintes características:

- É desenvolvido para um propósito específico, como sistemas automotivos de injeção eletrônica e ABS, brinquedos, câmeras e estufas. No caso do projeto desta monografia

ele tem o objetivo de monitorar o consumo de água. Porém, *smartphones* são considerados sistemas embarcados geralmente, mas desempenham uma variedade de funções distintas.

- Possuir *hardware* e *software* de menor custo, capacidade e funcionalidade. No geral são encontrados com configurações inferiores a PC's. Porém com o advento de novas tecnologias, essa característica está se tornando pouco descritiva.
- Possuir sistemas de maior confiabilidade e qualidade em comparação a outros sistemas computacionais. Em grande parte das aplicações que exijam especificidade existe um requisito de maior confiabilidade. Por exemplo, caso uma unidade de controle automotivo falhe durante a injeção eletrônica ou caso algum aparelho eletrônico em um centro cirúrgico também produza algum erro, o resultado pode ser catastrófico. Porém há aparelhos considerados sistemas embarcados que falhas não causam tanto dano, como aparelhos celulares.

2.5 Considerações finais

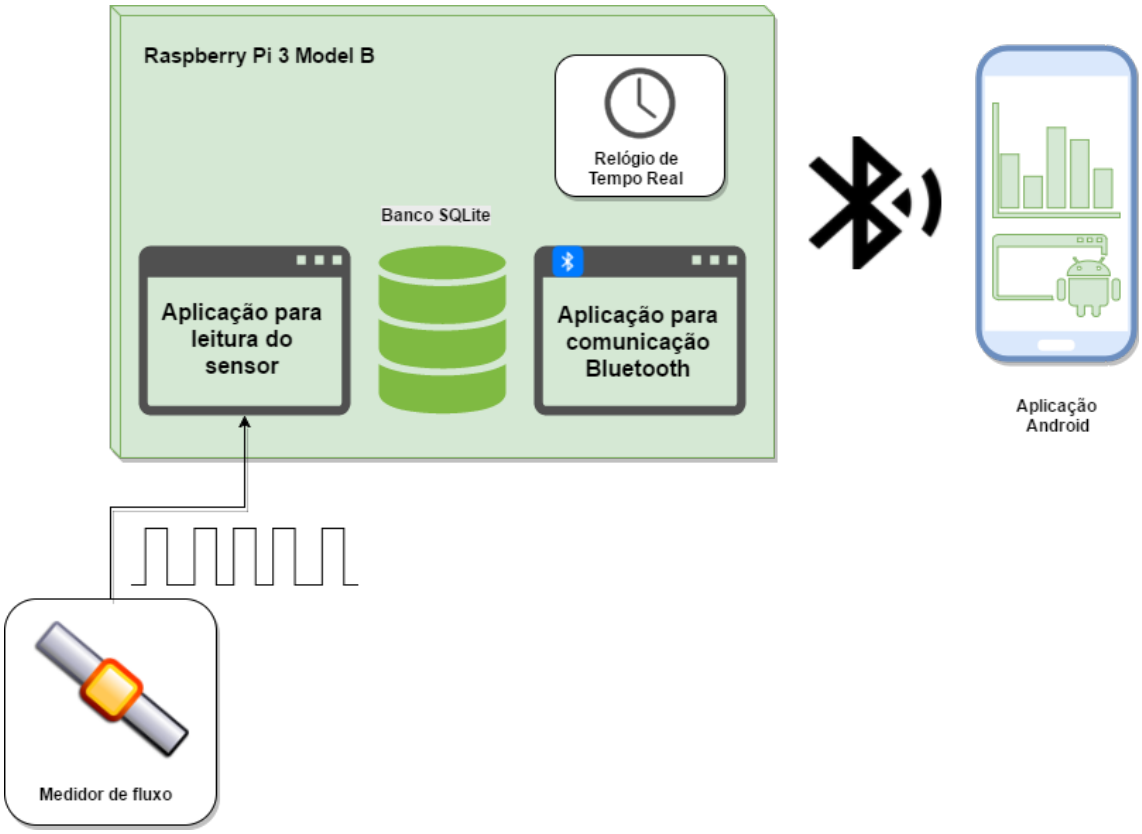
Nessa seção, foram apresentados os conceitos e técnicas que foram utilizados no desenvolvimento do projeto, os quais são fundamentais para o entendimento no decorrer do trabalho. No próximo capítulo é apresentado como o projeto foi desenvolvido e o funcionamento dos algoritmos utilizados.

3 DESENVOLVIMENTO DO TRABALHO

3.1 Projeto

Neste capítulo serão tratados os materiais, métodos e decisões que foram utilizados para a realização do projeto. Uma visão geral do projeto ilustrada na Figura 9, onde um maior detalhamento sera fornecido no decorrer do capítulo.

Figura 9: Visão geral do projeto



Fonte: Figura produzida pelo autor.

O projeto possui um medidor de vazão para realizar a medição da quantidade de líquido e os dados coletados são lidos pelo sistema embarcado utilizando um *script* de leitura dedicado a essa tarefa. A inicialização desse *script* depende do usuário, que escolhe quando as leituras se iniciam ou terminam. O *script* de leitura envia os dados para um banco de dados no sistema, que possui além da leitura do sensor os históricos de leituras feitas pelo usuário. O sistema embarcado também possui um *script* de comunicação a com um dispositivo móvel. O *script* de comunicação Bluetooth conecta-se com o aplicativo móvel e recebe diferentes comandos para enviar de diferentes formas os dados presentes no banco de dados para o aplicativo. Quando o sistema é energizado são iniciados o *watchdog*,

a biblioteca de leitura dos pinos de entrada e saída e o *script* de comunicação Bluetooth. O aplicativo do dispositivo móvel envia diversos comandos para o sistema embarcado, a fim de realizar diversas requisições no banco de dados. Essas requisições podem ser sobre leituras, histórico de consumo ou acompanhamento do consumo em tempo real. Ele envia também comandos para iniciar e finalizar o *script* de leitura e desligar o sistema. As aplicações implementadas no sistema embarcado foram feitos em Python e o aplicativo Android feito em Java. Todos os *scripts* desenvolvidos e o Aplicativo Android estão disponíveis nos repositórios do GitHub a seguir:

- https://github.com/lzanuzzo/TCCLucasTavaresZanuzzo
- https://github.com/lzanuzzo/TCCAppAndroid

3.2 Medidor de vazão e calibração

O medidor de vazão utilizado para o projeto é o YF-S201 ilustrado na Figura 10, com a especificação técnica no Anexo A. Esse medidor é facilmente encontrado no mercado nacional, possui o menor custo de aquisição e possui os suportes de rosca de ½ polegada para serem anexados na linha do cano. O medidor possui 3 fios de conexão: um vermelho de alimentação (5-24V DC), 1 preto (Terra) e um amarelo de saída para o sinal PWD, emitido pelo sensor de efeito Hall interno.

Figura 10: Medidor de vazão YF-S201

Fonte: Imagem produzida pelo autor

Dentre as principais características do medidor relevantes ao projeto estão, a vazão máxima de 30 l/min e uma pressão máxima de 2MPa. São relevantes, porque é importante para o projeto que o medidor possa operar em um hidrômetro residencial para controlar água de toda uma residência.

Será tomado como base o Manual de Normas técnicas da Sabesp (SABESP, 2012) para determinar as características básicas de funcionamento dos hidrômetros. A Sabesp é a empresa brasileira que detém a concessão dos serviços públicos de saneamento básico no Estado de São Paulo (SABESP, 2017). No manual consta-se que a NTS 161 padroniza os cavaletes de água em 1,5 m^3/h de vazão nominal, ou seja 25 l/min.

O medidor utilizado, segundo a documentação, suporta até uma vazão de 30 l/min, portanto está apto para funcionar nas condições normais de uma residência. Devido a imprecisão dos componentes e a imperfeição no processo de fabricação, uma calibração no medidor é necessária. Na calibração, será determinado o coeficiente que relaciona o número de voltas da hélice com a quantidade de água que passa através do medidor. Segundo a documentação do fabricante, a quantidade de pulsos por litro é de 450.

Para a calibração, foram feitas 14 medidas para 500ml, 1000ml, 1500ml e 2000ml utilizando um *script* em Python que contava apenas os pulsos do sensor. A partir de uma média dos valores encontrados, descritos na Tabela 1, foi possível determinar a quantidade de pulsos por litro do sensor. Os resultados apontam um erro de 6,88% no coeficiente e um coeficiente de 419 pulsos por litro.

Tabela 1: Resultados da calibração do medidor de vazão.

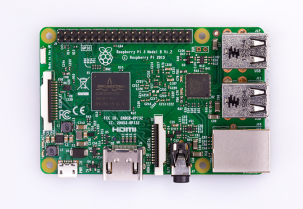
Pulsos para 500ml	Pulsos para 1000ml	Pulsos para 1500ml	Pulsos para 2000ml
188	422	623	881
204	421	620	838
226	427	616	857
210	436	616	845
227	387	625	848
235	412	623	848
226	407	620	860
207	403	621	775
220	411	625	845
201	411	638	862
203	429	630	846
205	412	628	855
203	442	575	864
200	398	634	862
Médias Parciais 211	415	621	849
Média Total 419			

Fonte: Produzido pelo autor.

3.3 Sistema embarcado

O sistema embarcado utilizado no projeto foi a Raspberry Pi 3 Model B, ilustrada na Figura 11. O sistema é de baixo custo, em torno de US$35, e também de tamanho muito reduzido. É um produto que pode ser facilmente adquirido e tem alto poder de processamento, com uma CPU 1.2GHz 64-bit quad-core ARMv8. Para o projeto, a plataforma possui outras características importantes como o hardware Bluetooth. Na placa foi instalado um sistema operacional Linux, Raspbian Lite, que é *open source* (RASPBERRY PI FOUNDATION, 2017c). O Raspbian já contém alguns pacotes pré instalados que são utilizados no projeto, como o banco de dados SQLite e o Python que também são *open source*.

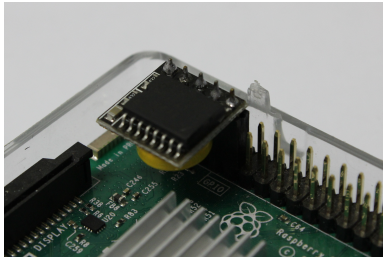
Figura 11: Raspberry Pi 3 Model B



Fonte: (RASPBERRY PI FOUNDATION, 2017b)

Esse modelo não possui um RTC integrado, e para que o projeto funcione *offline*, foi utilizado o *real time clock* DS3231. O DS3231, ilustrado na Figura 12, é um relógio de tempo real de alta precisão, para manter a data correta quando o dispositivo estiver sem alimentação. A data e hora corretas são importantes devido a necessidade de saber o momento exato das leituras no medidor para o usuário. O RTC é conectado e energizado nos pinos GPIO (RASPBERRY PI FOUNDATION, 2017a) da Raspberry e comunica-se utilizando I^2C, que é um barramento utilizado para conectar circuitos integrados. No caso do projeto, os pinos da plataforma são configurados para funcionarem como um barramento de comunicação de 2 vias entre o RTC e o sistema embarcado (NXP SEMICONDUCTORS, 2014) .

Figura 12: RTC DS3231



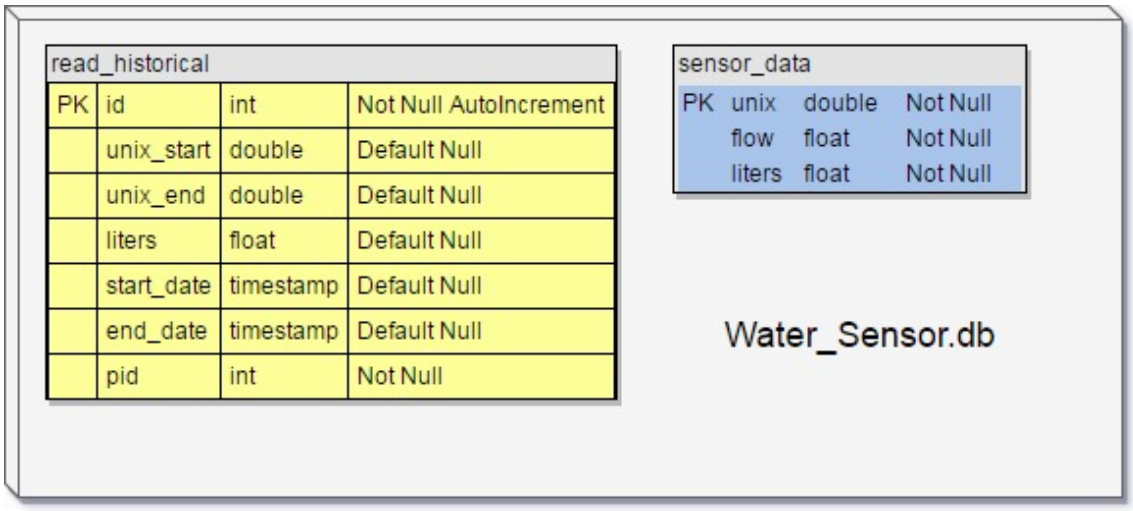
Fonte: Figura produzida pelo autor

3.4 Banco de dados

Para organizar e manipular a grande quantidade de dados criada pela aplicação, foi utilizado um banco de dados em SQLite, que é o mecanismo de banco de dados mais usado atualmente. O SQLite é uma biblioteca em C que implementa um banco de dados baseado em SQL, que é a linguagem de consulta padrão para bancos de dados relacionais. A principal característica que levou a utilização desse banco no projeto, foi o fato de que o SQLite é baseado em arquivos e não necessita de um processo servidor para manipular o banco de dados, como o projeto trabalha *offline* isso se torna um pré requisito. O SQLite também possui outras características vantajosas como, ter poucas dependências para uso, utilizando apenas bibliotecas C de arquivos e também não necessita de configurações de servidor. Mesmo não possuindo um SGBD, a biblioteca lida com alocações de memória e erros de I/O, mantendo as transações ACID (acrônimo de Atomicidade, Consistência, Isolamento e Durabilidade) mesmo com as falhas do sistema ou quedas de energia (SQLITE, 2017).

Para o projeto, foram criadas duas tabelas, a primeira tabela contém todos os dados coletados pelos sensor, com o tempo em *unix time* como chave primária, o valor da vazão em l/min e litros em ml. O *unix time* é uma medida de tempo baseada na quantidade de segundos passados desde 1 de janeiro de 1970(UTC). Essa tabela será utilizada para guardar os valores de todas as leituras. A segunda tabela armazena o histórico de leituras feitas pelo usuário. A tabela possui o intervalo de tempo de início e fim de cada leitura em *unix time* e em data romana, o valor total em litros da leitura e o ID do processo que executou aquela leitura. A estrutura das tabelas está apresentada na Figura 13

Figura 13: Estrutura do banco de dados utilizado no projeto



Fonte: Figura produzida pelo autor.

3.5 Script de leitura do sensor

O *script* de leitura do sensor é o responsável por coletar os dados lidos no sensor e armazenar no banco de dados. Esse *script* é iniciado quando o usuário deseja realizar uma leitura do consumo de água e é finalizado quando deseja-se finalizar a leitura. Quando iniciado, o *script* realiza as configurações do sensor, realiza a conexão com o banco de dados e verifica qual foi a última leitura feita, pois pode existir o caso em que o *script* por algum motivo foi finalizado indevidamente e a leitura não foi finalizada. Caso não exista nenhuma leitura inacabada, ele insere no banco de dados os parâmetros iniciais da leitura, como a data de início e o ID do processo. Caso exista uma leitura inacabada, ele sobrescreve o ID de processo antigo e retoma a leitura inacabada a partir do ponto que parou.

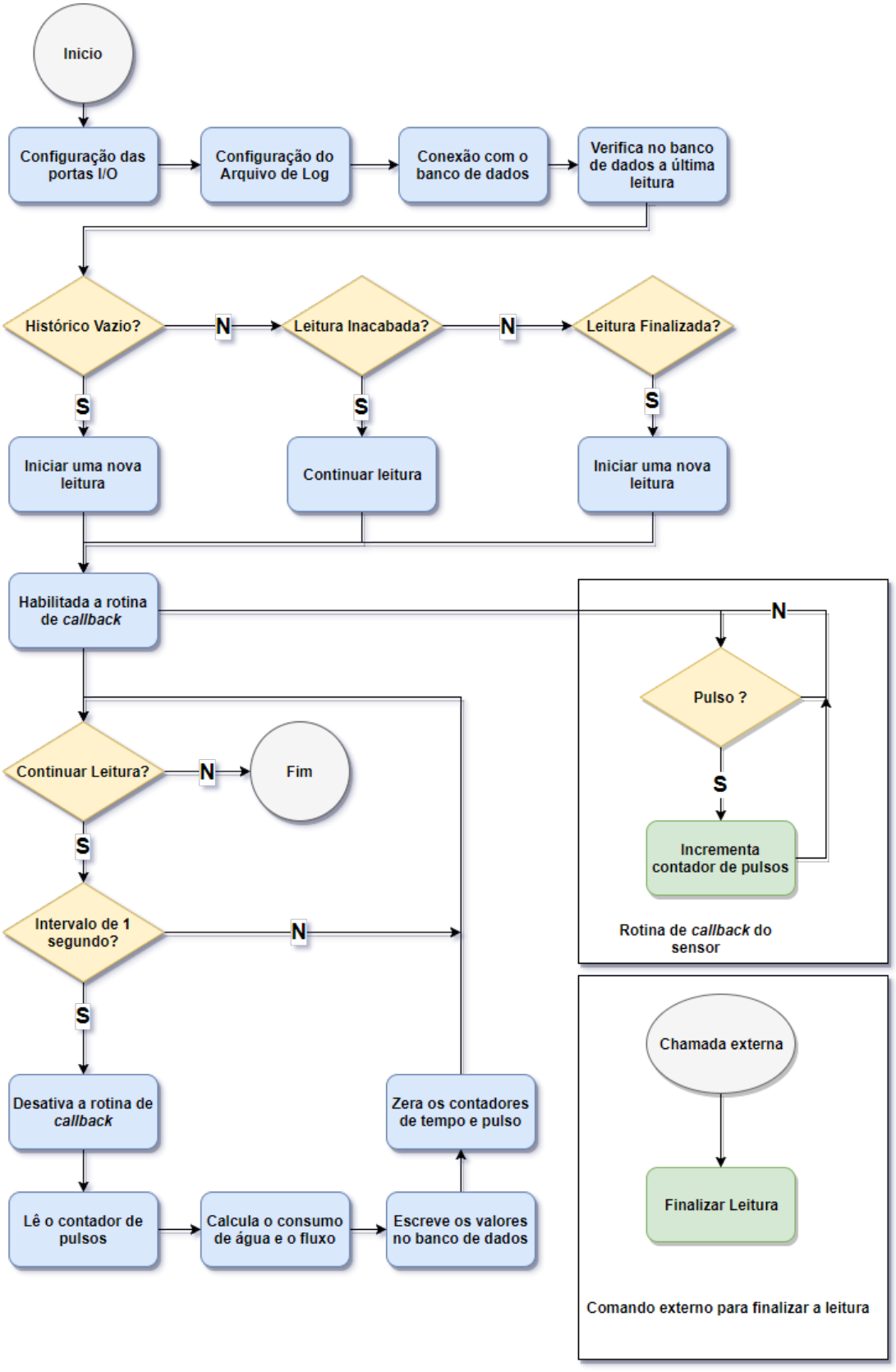
Posteriormente, o *script* entra no laço principal. A cada intervalo de 1 segundo o *script* verifica quantos pulsos ocorreram no sensor e escreve no banco o tempo da leitura, a quantidade de líquido e a vazão nesse intervalo de tempo. Os pulsos do sensor são armazenados em uma variável que é incrementada a cada chamada de uma função de *callback*. A função de *callback*, é uma rotina chamada sempre que é detectada uma borda de subida no pino de I/O associado ao sensor. A Figura 14 ilustra o funcionamento do *script*. Para finalizar o processo atual do *script*, o usuário envia um comando em *bash* que envia uma mensagem ao processo e o finaliza. Antes de finalizar o *script*, é inserido no banco o total de litros atual da leitura e o momento em que a leitura foi finalizada.

3.6 Script de comunicação com o aplicativo

O *script* de comunicação com o aplicativo, é o responsável por conectar-se ao dispositivo Android por Bluetooth, para enviar dados e receber comandos. O *script*, ao iniciar-se, espera por alguma conexão Bluetooth. Após realizada a conexão, ele entra em um estado inicial, responsável por enviar os dados relativos a leitura atual. O *script* possui um laço principal, onde são enviados dados constantemente e a cada iteração do laço principal, ele espera um comando vindo por Bluetooth do dispositivo móvel.

A função para receber a mensagem Bluetooth vinda do dispositivo é bloqueante e possui um *countdown*. Esse *countdown* é definido para cada estado do aplicativo e é usado como intervalo entre as mensagens que o *script* envia, ou seja, a espera do laço. Por exemplo, o intervalo de tempo para enviar os dados da leitura atual é de 1 segundo, pois essa é a taxa de aquisição do *script* que lê os dados do sensor. E então, a cada iteração do laço ele espera por 1 segundo antes de enviar o próximo dado. Caso a conexão com o aplicativo seja finalizada por algum motivo, ele continua ativo e uma função bloqueante aguarda uma nova conexão Bluetooth. Cada comando e ação serão exibidos no fluxograma da Figura 15, e cada estado do *script* explicado em detalhes abaixo.

Figura 14: Diagrama de fluxo do *script* de leitura do sensor.



Fonte: Figura produzida pelo autor.

Leitura atual: Para enviar a leitura atual, o *script* consulta a cada segundo (mesma taxa de escrita do medidor no banco) o último valor inserido no banco de dados e envia por Bluetooth para o aplicativo, caso uma leitura estiver acontecendo os dados serão enviados também para o aplicativo. Caso nenhuma leitura esteja ocorrendo, ele enviará mesmo assim o último valor do banco, que exibe em que ponto a última leitura foi finalizada.

Iniciar leitura: Quando o *script* recebe um comando para iniciar uma leitura, ele envia um comando *shell* para executar o *script* de leitura. O *script* de leitura lidará com eventuais problemas de conflito caso o usuário tente iniciar duas leituras simultâneas.

Finalizar leitura: Quando o *script* recebe um comando para finalizar uma leitura, ele envia um comando *shell* para finalizar o *script* de leitura. Uma consulta no banco de dados verifica se existem leituras correntes e então a partir do *process* ID do *script* de leitura a ser finalizado, uma mensagem é recebida para finalizar o *script* de leitura.

Enviar histórico de leituras: Quando o *script* recebe um comando para enviar o histórico de leituras, este consulta todas as informações do histórico no banco de dados e salva em uma lista que será enviada para o dispositivo, elemento por elemento no laço principal. Marcadores são colocados no início e no fim da lista para o aplicativo poder definir corretamente o início e o fim do histórico.

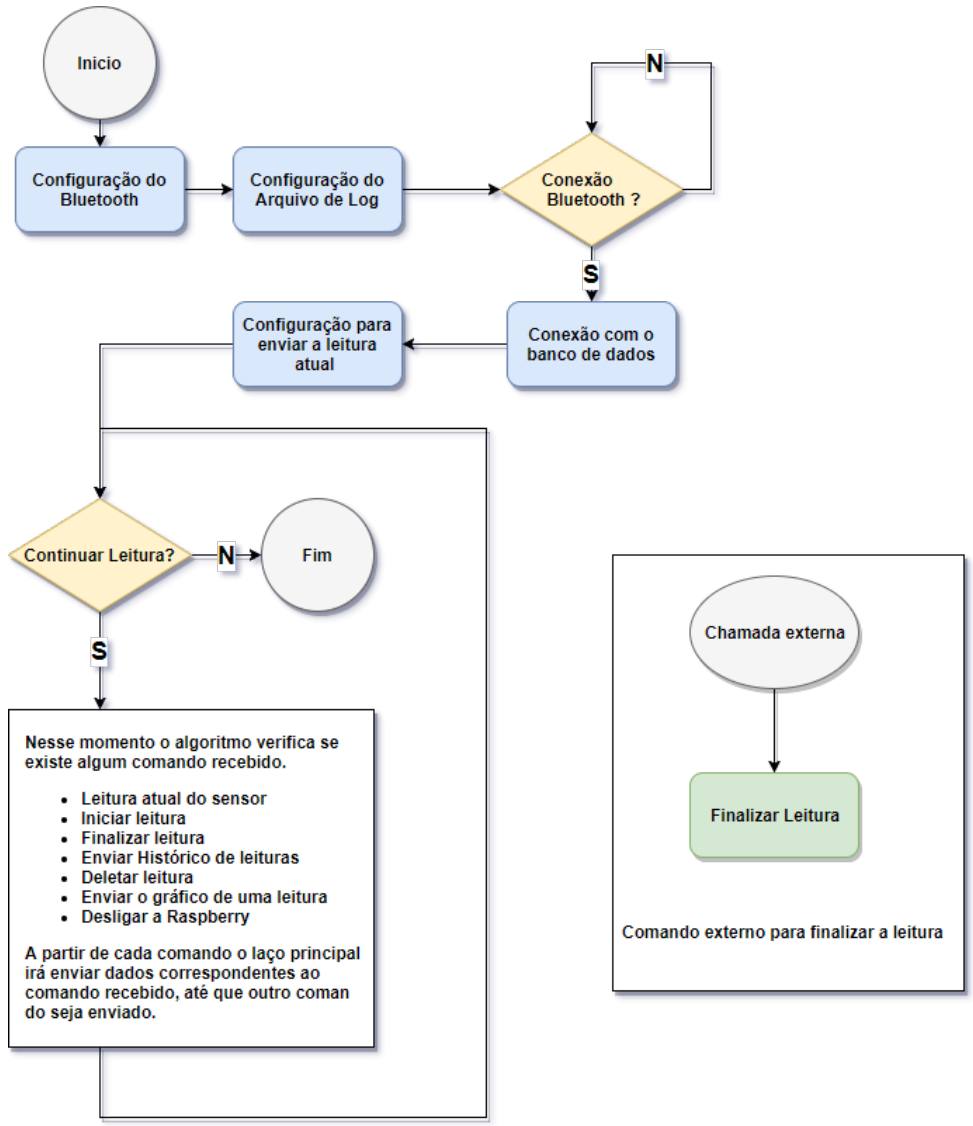
Deletar uma leitura: Quando o *script* recebe um comando para deletar uma leitura do banco de dados, este recebe também o ID da leitura a ser deletada. A partir do ID a leitura é deletada do banco de dados, e armazena-se o inicio e o fim dessa leitura. Com o inicio e o fim da leitura deletada, o *script* deleta esse intervalo de dados da tabela de dados do sensor.

Enviar o gráfico de uma leitura: Quando o *script* recebe um comando para retornar o gráfico de determinada leitura para ser exibido, ele também recebe o ID da leitura a ser enviada. A partir do ID, o *script* realiza uma consulta para saber quantas entradas existem para uma determinada leitura. Algumas leituras podem acabar ficando com uma carga muito grande de dados para serem retornados rapidamente para o usuário. E também não acrescentaria nada na resolução do gráfico uma quantidade grande de pontos que não influenciam na resolução. O algoritmo salva em uma lista no máximo 100 valores, resolução suficiente para um gráfico nessas dimensões. Por exemplo, caso existam 1000 entradas para o gráfico, é selecionado um ponto a cada 10 entradas. Marcadores são colocados no início e no fim da lista, para o aplicativo poder definir corretamente o início e o fim do gráfico para exibir na tela.

Desligar a Raspberry: Quando o *script* recebe um comando para desligar a Raspberry, um comando *shell* para desligar a Raspberry é executado. É recomendado desligar

corretamente o sistema operacional. Desligar diretamente na alimentação pode ocasionar, por exemplo, a corrupção do sistema de arquivos.

Figura 15: Diagrama de fluxo do *script* de comunicação com o aplicativo.



Fonte: Figura produzida pelo autor.

3.7 Aplicativo Android

O aplicativo foi desenvolvido em Java para o sistema operacional Android. O dispositivo deve possuir um sistema operacional Android igual ou superior a KitKat 4.4.4 (API *level* 19) ou superior. Também deve possuir a estrutura para se comunicar com o sistema embarcado usando Bluetooth. O aplicativo para o projeto possui as seguintes funções:

- Visualização da aquisição de dados corrente no sensor, com alguns parâmetros.

- Possibilidade de edição de alguns parâmetros. Os parâmetros editáveis permitem fazer um cálculo do custo aproximado da água consumida e determinar uma meta de consumo.
- Visualização das leituras em formato de histórico em uma lista.
- Visualização das leituras em formato gráfico.
- Excluir as leituras do histórico.
- Desligar o sistema embarcado.

3.7.1 Classe Principal

A classe principal do aplicativo, ou tela inicial ilustrada na Figura 17, é a qual será possível visualizar a leitura corrente. Também possibilita o acesso as outras funcionalidade do aplicativo a partir de um menu lateral .

Ao iniciar o aplicativo, uma janela irá questionar se é possível habilitar o Bluetooth, caso ele não esteja ativado. Caso o usuário confirme a ativação, o aplicativo irá iniciar, caso contrário ele será finalizado. Ele será finalizado, pois não possui nenhuma funcionalidade sem o Bluetooth habilitado. Após iniciar o adaptador Bluetooth, o aplicativo irá procurar os dispositivos disponíveis para parear. Caso pareie com sucesso com a Raspberry, ele irá criar um *socket* de conexão e então um comando é enviado a Rasbperry para enviar os dados de leitura do banco de dados. Caso não encontre-a dentre os dispositivos disponíveis, uma notificação será exibida.

Durante a instanciação da atividade também é consultado um arquivo local, criado durante a primeira execução do aplicativo, com os parâmetros de tarifa, meta de consumo e mínimo consumo. Caso esse arquivo não exista, ele é criado e são colocados os parâmetros padrão.

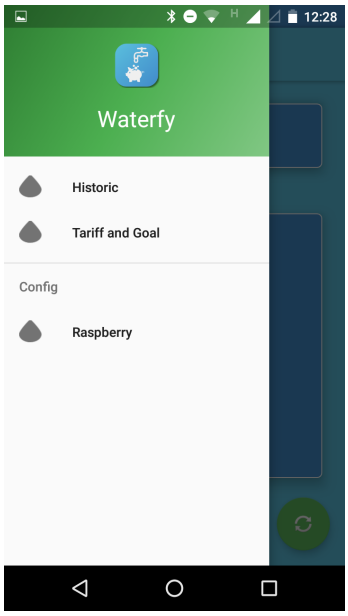
Na parte superior da tela principal observada na Figura 17, com o título do aplicativo, encontra-se o botão que ativa o menu, observado na Figura 16. Para cada item do menu, existe um *listener* que chama a sua respectiva classe. No *container* mais abaixo, está o status da comunicação entre a Raspberry e o dispositivo. Ele possui 3 status possíveis:

***No Device*:** Ele exibirá esse status junto com um alerta em vermelho para indicar que não encontrou a Raspberry para o pareamento.

Paired: Ele exibirá esse status quando o dispositivo estiver com permissão para requisitar uma conexão com a Raspberry, juntamente com o simbolo em vermelho. Estar pareado não significa que a conexão seja possível, é uma condição necessária mas não suficiente.

Synchronized: Ele exibirá esse status quando uma conexão com a Raspberry foi realizada com sucesso e o simbolo se tornará verde.

Figura 16: Menu lateral do aplicativo



Fonte: Figura produzida pelo autor.

Abaixo do *status* da conexão, está o *container* com os dados. Ele exibe os valores correntes da leitura, os parâmetros utilizados para o cálculo do gasto com o consumo, o total gasto até o momento e a meta de consumo. Eles são atualizados a partir da *thread* iniciada pelo botão verde de *sync*.

Na parte mais baixa da tela ilustrada na Figura 17, temos 3 botões, suas funções são detalhadas a seguir:

Botão verde de sincronismo: Esse botão irá sincronizar e conectar o aplicativo com a Raspberry e exibirá os dados correntes da leitura. Caso não esteja ocorrendo nenhuma leitura, ele irá mostrar em que ponto a última leitura terminou. Quando o botão é pressionado, uma *thread* é iniciada e verifica-se a existência de um *socket* entre a Raspberry e o dispositivo, então é declarado um canal de entrada para receber o *stream* de *bytes* enviado pela Raspberry. Quando o canal é estabelecido com sucesso, o recipiente com o status da conexão é atualizado para “Synchronized” e um indicador verde é exibido no lugar do vermelho onde antes estava escrito “Paired”. Enquanto existirem bytes sendo enviados pela Raspberry, um laço mantém atualizados os dados no recipiente da tela. O conjunto de bytes recebidos são convertidos em caracteres ASCII e a partir de delimitadores pré definidos, consegue-se determinar os valores que serão exibidos para o usuário.

Botão END READ Esse botão irá enviar uma mensagem para a Raspberry finalizar a leitura corrente, ou seja, finalizar o *script* de leitura em execução no momento. Caso não esteja ocorrendo nenhuma leitura ele não causa nenhuma ação. O aplicativo inicia uma tarefa assíncrona que, após verificar a existência de um *socket* de comunicação, cria um canal para enviar dados. Após a criação do canal, um comando para finalizar o *script* de leitura é enviado. Caso o *socket* não exista ou o canal de saída não puder ser criado, uma notificação será exibida.

Botão BEGIN READ Esse botão irá enviar uma mensagem para a Raspberry e iniciar uma leitura, ou seja, iniciar o *script* de leitura e para exibir os dados na tela. Caso já esteja ocorrendo uma leitura ele não causa nenhuma ação. O aplicativo inicia uma tarefa assíncrona que, após verificar a existência de um *socket* de comunicação, cria um canal para enviar dados. Após a criação do canal, um comando para iniciar o *script* de leitura é enviado. Caso o *socket* não exista ou o canal de saída não puder ser criado, uma notificação será exibida.

Figura 17: Classe Principal do aplicativo

Fonte: Figura produzida pelo autor.

3.7.2 Classe Histórico

A classe exibe uma lista com todas as leituras feitas que estão presentes no banco de dados da Raspberry. Cada item da lista exibe para o usuário o ID da leitura, definido automaticamente pelo banco de dados, data do início, data do fim, total em litros e o gasto em reais baseado na configuração atual.

A metade superior da tela de histórico é um recipiente para inserir o gráfico das leituras e possui algumas instruções no recipiente para a classe. Ao clicar em algum item

de leitura, o gráfico respectivo será exibido e após um longo clique é possível excluir a leitura. O gráfico exibido no aplicativo foi feito utilizando a biblioteca Androidplot (ANDROIDPLOT, 2017) e é compatível a partir do Android 1.6.

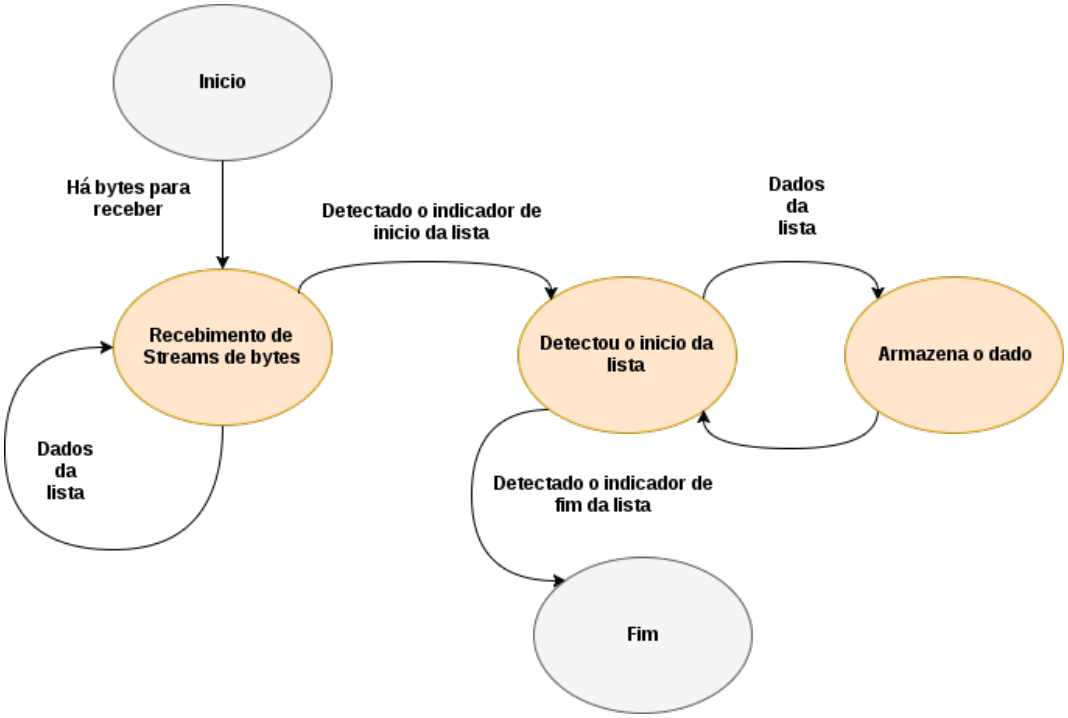
Figura 18: Classe Histórico, tela inicial

Fonte: Figura produzida pelo autor.

A atividade inicia-se verificando se o adaptador Bluetooth está ativado, o aplicativo irá procurar os dispositivos disponíveis para parear. Caso encontre a Raspberry, ele irá criar um *socket* de conexão e então cria-se um canal para enviar dados. Após a criação do canal é enviado um comando para a Raspberry enviar o histórico de leitura do banco de dados. Caso não encontre a Raspberry dentre os dispositivos disponíveis, uma notificação será exibida e a atividade será finalizada e retornará para a classe principal.

Após o sucesso do envio do comando, uma função é chamada para coletar os dados do histórico enviados pela Raspberry. Essa função é uma tarefa assíncrona que quando iniciada, cria um canal de comunicação com a Raspberry usando o *socket*. A tarefa entra em um laço que verifica se *bytes* estão sendo recebidos e detecta o conjunto de dados correspondentes ao histórico. Como os dados do histórico são enviados em um laço pela Raspberry, é necessário identificar o início e o fim da *stream*. A Figura 19 ilustra o algoritmo utilizado no processo e quando todos os dados relativos ao histórico são coletados dessa forma, eles são exibidos no aplicativo.

Figura 19: Fluxograma do algoritmo que recebe a stream de dados

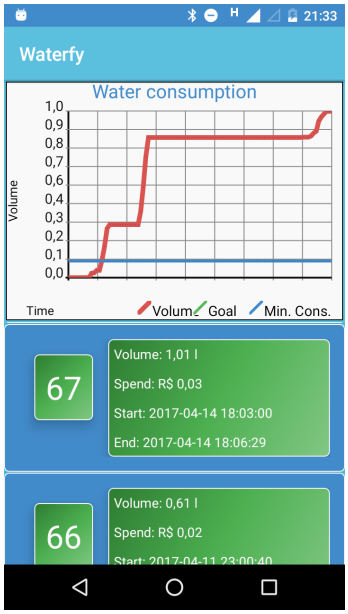


Fonte: Figura produzida pelo autor.

Quando o usuário clica em algum registro do histórico, o gráfico dessa leitura vai ser exibido como ilustrado na Figura 20. Quando ocorre o clique é iniciada uma tarefa assíncrona que recebe o ID da leitura selecionada. Essa função cria um canal de comunicação com a Raspberry usando o *socket* da atividade e então envia o comando para a Raspberry retornar o gráfico juntamente com seu ID. Caso o envio do comando tenha tido sucesso, uma nova tarefa assíncrona é criada para ler os dados do gráfico. A tarefa entra em um laço que verifica se *bytes* estão sendo recebidos e detecta o conjunto de dados correspondentes ao gráfico, seguindo mesmo algoritmo da Figura 19

Quando o usuário realiza um clique longo em uma leitura, uma janela irá aparecer para confirmar a exclusão da leitura. Em caso positivo, é iniciada uma tarefa assíncrona o ID da leitura selecionada. Essa função cria um canal de comunicação com a Raspberry usando o *socket* da atividade e então envia o comando para a Raspberry para excluir a respectiva leitura.

Figura 20: Classe Histórico, gráfico apresentado



Fonte: Figura produzida pelo autor.

3.7.3 Classe Meta e Tarifa

Essa atividade, ilustrada pela Figura 21, representa os parâmetros configuráveis pelo usuário:

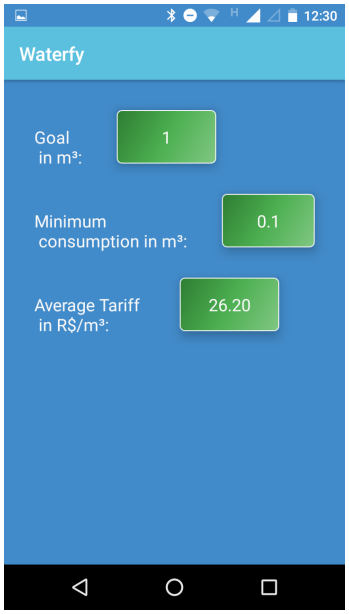
Goal in m³: O valor para a linha de meta que será exibida no gráfico do histórico.

Minimum consumption in m³: O valor minimo de consumo para iniciar a cobrança.

Average Tariff in R$/m³: Tarifa média de consumo.

Esses três valores são armazenados em um arquivo. Durante a inicialização da atividade é verificada a existência desse arquivo e então os dados são lidos e mostrados na tela. Quando o usuário muda qualquer um desses valores pelo aplicativo, uma função é chamada e escreve o novo valor no arquivo.

Figura 21: Classe Meta e Tarifa



Fonte: Figura produzida pelo autor.

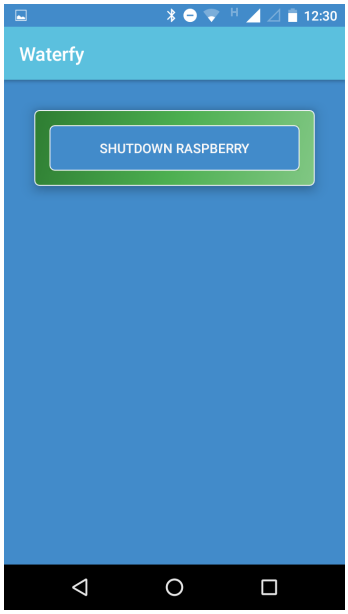
3.7.4 Classe Raspberry

Essa atividade é responsável por permitir que a Raspberry seja desligada com segurança e para tal ação é utilizado um simples botão, como ilustrado na figura Figura 22. A atividade inicia-se verificando se o adaptador Bluetooth está ativado, o aplicativo irá procurar os dispositivos disponíveis para parear. Caso encontre a Raspberry, ele irá criar um *socket* de conexão e então cria-se um canal para enviar dados. Após a criação do canal, é enviado um comando para a Raspberry executar o comando que desliga o sistema operacional de forma segura.

3.7.5 Classe Gráfico

Essa classe não representa nenhuma tela no aplicativo, ela é um fragmento responsável por anexar o gráfico no histórico. Nessa classe são configuradas as características do gráfico e determinados os pontos a serem desenhados na curva. São definidas também características como título, legenda, eixos e suavidade das curvas.

Figura 22: Classe Raspberry do aplicativo



Fonte: Figura produzida pelo autor.

3.8 Considerações finais

Nesse capitulo foi apresentado todo o desenvolvimento do projeto, desde a calibração do medidor e a descrição do *hardware* utilizado quanto os algoritmos e a implementação do aplicativo. No próximo capítulo será validado todo o sistema proposto até agora através de alguns testes.

4 TESTES, RESULTADOS E DISCUSSÕES

Para testar o projeto, foi realizada uma coleta controlada de água e seu valor foi comparado com o valor armazenado pelo sistema. O sistema foi instalado em um encanamento com uma torneira para controlar a passagem de água, ilustrado na Figura 23.

Sendo assim, o sistema foi energizado e usando o aplicativo e com a torneira fechada uma leitura foi iniciada, como na Figura 24. Para saber a quantidade de água que saiu da torneira, um recipiente onde está determinado a quantidade de 2 litros foi utilizado. Com a leitura iniciada, a torneira foi aberta e o recipiente cheio até a marca de 2 litros. Com os 2 litros completos a torneira foi fechada e a água descartada. O processo foi repetido 15 vezes e então a leitura foi finalizada, totalizando 30 litros de água.

Figura 23: Instalação do medidor de vazão

Fonte: Figura produzida pelo autor.

Figura 24: Sistema de testes

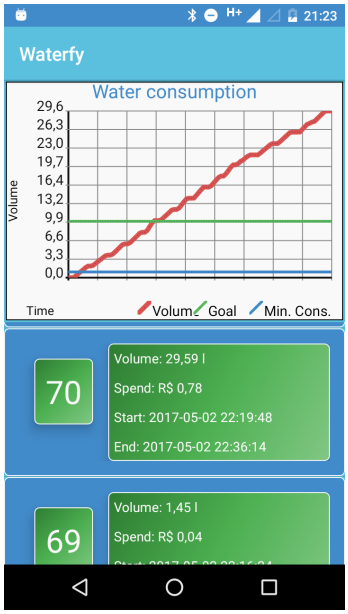
Fonte: Figura produzida pelo autor.

O resultado do teste pelo aplicativo, ilustrado na Figura 25, mostra que foram lidos

aproximadamente 29,6 litros de água dos aproximados 30 litros que passaram pela torneira, apresentando um desempenho promissor. Temos o fator do erro humano na medição, pois a torneira não era exatamente fechada no preciso instante e também há um provável erro na marcação de 2 litros do recipiente.

O mesmo teste foi feito com o sensor descalibrado e o resultado está apresentado na Figura 26. Nota-se uma grande defasagem no valor encontrado, 25,6 litros lidos pelo sistema contra aproximados 30 litros que passaram pela torneira.

Figura 25: Teste com o medidor calibrado

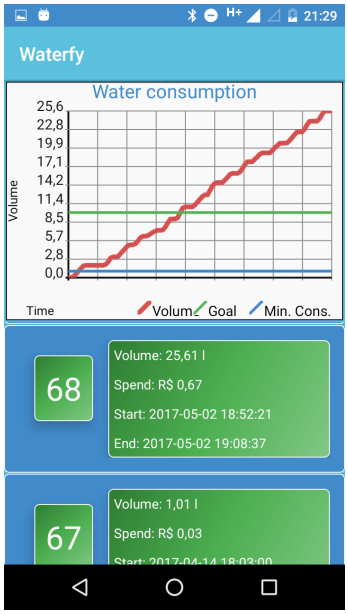


Fonte: Figura produzida pelo autor.

Um segundo teste foi realizado utilizando uma máquina de lavar roupas. O sistema foi instalado na entrada de água da máquina de lavar para medir o consumo de água durante uma lavagem. A máquina utilizada é uma Consul modelo: CWI07A de 7kg, e suas informações técnicas estão disponíveis no Anexo B.

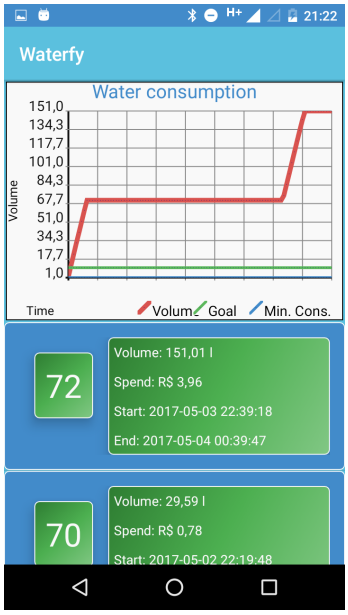
Para a configuração Normal com água no Nível 4, segundo o manual, temos um consumo de água de aproximadamente 144 litros. Então com a máquina energizada em 127V uma leitura foi iniciada e após isso a máquina vazia foi ligada na configuração Normal com água no nível 4. Após a lavagem a leitura é finalizada o resultado da leitura é ilustrado na Figura 27, que mostra que o sistema encontrou um valor de 151 litros. Na Figura 27 é visível o momento do enchimento no inicio e o enxágue mais ao fim.

Figura 26: Teste com o medidor descalibrado



Fonte: Figura produzida pelo autor.

Figura 27: Teste com o máquina de lavar



Fonte: Figura produzida pelo autor.

4.1 Considerações finais

Os testes realizados com sistema apontam resultados promissores e a GUI atuou como uma forma interativa de controle e visualização. O próximo capitulo irá retomar o que foi feito no projeto e apresentará alguns próximos passos possíveis para o atual sistema.

5 CONCLUSÃO

A proposta do trabalho foi projetar um sistema de monitoramento do consumo de água em um aplicativo Android e os resultados foram promissores. Os resultados obtidos no Capítulo 4 foram consistentes e apresentaram uma faixa de erro pequena. O projeto se comportou de maneira adequada para aplicações em um ambiente residencial. Porém, para uma melhor validação do sistema seria necessário mais testes, como por exemplo, atuar na linha de cano do hidrômetro por um longo período de tempo.

O aplicativo atende aos requisitos propostos e permite ao usuário o controle sobre as leituras em uma interface amigável. O uso de *smartphones* facilita a utilização do sistema, pois é uma plataforma já conhecida e de muita familiaridade. Porém, a comunicação por Bluetooth atrapalhou um pouco a fluidez do funcionamento, fazendo com que o usuário esperasse alguns segundos por alguns comandos, devido ao tempo de sincronização.

O uso de um sistema operacional ajudou bastante no desenvolvimento do projeto. Ele ajudou no escalonamento de processos entre o *script* de leitura e o de comunicação Bluetooth. E também, permitindo o uso de bancos de dados e a comunicação Bluetooth. O uso de Bancos de Dados foi fundamental para a implementação, sua utilização permite que os dados sejam organizados de forma segura e seu acesso seja controlado pelo banco, impedindo qualquer inconsistência e gerenciando múltiplos acessos.

5.1 Trabalhos futuros

A partir do projeto atual serão apresentadas algumas vertentes possíveis para a sua evolução:

- Utilizando o sistema atual, o aplicativo móvel poderia ser estendido para uma rede social, onde as pessoas poderiam compartilhar seu consumo de água, incentivando ainda mais a sua utilização e a economia.

- O *script* para a leitura do sensor é uma parte modular do projeto e pode facilmente ser substituído. Ou seja, o projeto pode ser utilizado para a leitura de outros sensores que produzam séries temporais com pouquíssimas modificações no sistema. Por exemplo, leitura do consumo de energia elétrica, umidade ou temperatura.

- O sistema poderia utilizar um sensor que tolere maior vazão de água para trabalhar em uma gama maior de aplicações, como piscinas ou industrias.

- A interface do usuário poderia ser implementada em um sistema *web*, a implementação seria muito simplificada. Como o protocolo utilizado seria o HTTP a comunicação

seria muito mais segura, seria possível trabalhar com dados mais complexos e melhores SGBD's como MySQL. Um sistema *web* permitiria a criação de *dashboards* mais complexos e maior interação com o usuário. A conexão com a rede possibilitaria a remoção do RTC e manter o horário pela rede.

- O sistema poderia ser utilizado na educação de crianças e jovens. A utilização de uma tecnologia familiar pode ajudar a incorporar uma cultura de consumo consciente no inicio da vida.

REFERÊNCIAS

ALBERT, S. H. **Bluetooth Essentials for Programmers**. [S.l.: s.n.], 2007. 1-26 p. ISBN 9780521703758.

ANDROIDPLOT. 2017. Disponível em: <http://androidplot.com/>. Acesso em: 06/05/2017.

COHEN, O. **O fundo do poço da crise hídrica em São Paulo**. 2015. Disponível em: <http://exame.abril.com.br/brasil/o-fundo-do-poco-da-crise-hidrica-em-sao-paulo/>. Acesso em: 06/05/2017.

DURAO, V. S. **Pesquisa Mostra que 48% dos brasileiros não controlam o consumo de água**. 2012. Disponível em: <http://www.valor.com.br/brasil/2729060/pesquisa-mostra-que-48-dos-brasileiros-nao-controlam-consumo-de-agua>. Acesso em: 06/05/2017.

EDWARD, R. **Hall-Effect Sensor**. [S.l.]: ELSEVIER, OXFORD, 2006. 1-9 p. ISBN 978-0-7506-7934-3.

FOLHA DE SãO PAULO. **Número de smartphones em uso no Brasil chega a 168 milhões, diz estudo**. 2016. Disponível em: <http://www1.folha.uol.com.br/mercado/2016/04/1761310-numero-de-smartphones-em-uso-no-brasil-chega-a-168-milhoes-diz-estudo.shtml>. Acesso em: 06/05/2017.

KANTAR. **Kantar Brasil Insights**. 2017. Disponível em: <http://br.kantar.com/>. Acesso em: 27/06/2017.

_____. **Smarthphone OS sales market share**. 2017. Disponível em: <https://www.kantarworldpanel.com/global/smartphone-os-market-share/>. Acesso em: 26/06/2017.

NXP SEMICONDUCTORS. **2C-bus specification and user manual**. [S.l.], 2014. Disponível em: <http://www.nxp.com/documents/user_manual/UM10204.pdf>. Acesso em: 06/05/2017.

RASPBERRY PI FOUNDATION. **GPIO: RASPBERRY PI MODELS A AND B**. 2017. Disponível em: <https://www.raspberrypi.org/documentation/usage/gpio/>. Acesso em: 06/05/2017.

_____. **RASPBERRY PI 3 MODEL B**. 2017. Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>. Acesso em: 06/05/2017.

_____. **RASPBIAN**. 2017. Disponível em: <https://www.raspberrypi.org/downloads/raspbian/>. Acesso em: 06/05/2017.

ROSSI, M. **Crise hídrica no Estado de São Paulo: "Resta água para apenas 38 dias"**. 2014. Disponível em: <http://brasil.elpais.com/brasil/2014/09/26/politica/1411739708_069324.html>. Acesso em: 06/05/2017.

SABESP. **Norma Técnica Sabesp NTS 181**: Dimensionamento do ramal predial de água, cavalete e hidrômetro – primeira ligação. São Paulo, 2012. Disponível em: <http://www2.sabesp.com.br/normas/nts/NTS181.pdf>.

_____. **Sabesp**. 2017. Disponível em: <http://site.sabesp.com.br>. Acesso em: 06/05/2017.

SQLITE. **About SQLite**. [S.l.], 2017. Disponível em: <http://sqlite.org/about.html>. Acesso em: 06/05/2017.

TAMMY, N. **Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers**. [S.l.: s.n.], 2005. 5-16 p. ISBN 9780123821966.

THE INTERNET SOCIETY. **A Universally Unique IDentifier (UUID) URN Namespace.** [S.l.], 2005. Disponível em: <https://tools.ietf.org/html/rfc4122>. Acesso em: 10/05/2017.

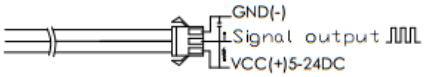
UPP, E. L. **Fluid Flow Measurement**: A practical guide to accurate flow measurement. [S.l.]: Butterworth–Heinemann, 2002. 154-212 p. ISBN 0-88415-758-X.

Anexos

ANEXO A – ESPECIFICAÇÕES TÉCNICAS DO MEDIDOR DE FLUXO YF-S201

YIFA the plastics Ltd
Prodcut Introduction

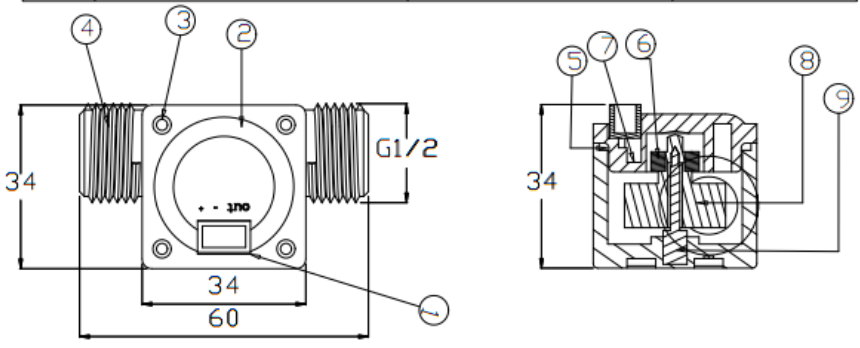
1.Modle:YF-21
2.Product Name:Hall sensor
3.Flow Range: 1-30L/MIN
4.(1)Connection Method



(2)Voltage Range 3.5-24VDC, Pluse Characteristic:F=7Q(L/MIN).
(3)Extent of error:±5%.
(4)Flow-Pulse
2L/MIN=16HZ 4L/MIN=32.5HZ 6L/MIN=49.3HZ
8L/MIN=65.5HZ 10L/MIN=82HZ

5.Bom

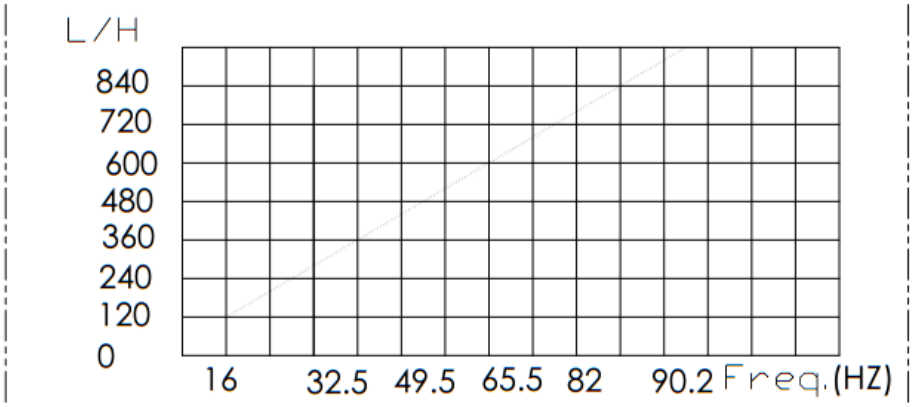
No.	Item	Material	Qty.
1	Connection wire		1
2	Bonnet	PA	1
3	Screw		4
4	Valve body	PA	1
5	Leak press valve		1
6	Magnet		1
7	Hall		1
8	Impeller	POM	1
9	Rustless steel axis	SUS304	1
10			
11			



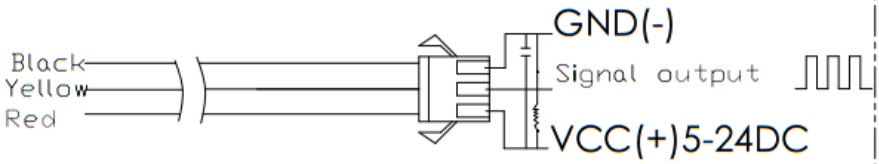
Flow-Pulse	Flow-Curve	Connection method

一.Flow Range:100L/H-/1800H-L/H

Flow (L/H)	Frezq.(HZ)	Erro range
120L/H	16	±10
240L/H	32.5	
360L/H	49.3	
480L/H	65.5	
600L/H	82	
720L/H	90.2	



Connection method:



ANEXO B – ESPECIFICAÇÕES TÉCNICAS DA LAVADORA CONSUL CWI07A

Dados Técnicos

7 kg CWI07A

Consul Floral

Dimensão e peso sem embalagem	
Altura com a tampa fechada (sem os pés)	103 cm
Altura com a tampa aberta (sem os pés)	138 cm
Largura	54,5 cm
Profundidade	61,0 cm
Peso	29,5 kg

Dimensão e peso com embalagem	EPS (isopor)	Papelão
Altura	105,5 cm	105,5 cm
Largura	58,0 cm	58,0 cm
Profundidade	66,0 cm	64,0 cm
Peso	30,5 kg	34,5 kg

Consumo máximo de energia	
127 V	0,12 kWh*
220 V	0,11 kWh*

Consumo máximo de água	
127 V	144 litros*
220 V	143 litros*

Capacidade de roupa seca	7 kg**
Centrifugação	480 rpm
Tensão	127 V~/220 V~
Variação de tensão admitida	106 a 132 V~/198 a 242 V~
Freqüência	60 Hz

Intensidade de corrente	
127 V	6,3 A
220 V	3,3 A

Potência máxima	
127 V	755 W
220 V	710 W

** Programa Lavagem Normal, Nível 4 de água.*
*** Capacidade baseada na carga padrão descrita na Portaria do Ministério do Desenvolvimento, Indústria e Comércio Exterior - MDIC, nº 185*

Fale com a Consul

SAB ÚNICO AUTORIZADO CONSUL

Para tirar dúvidas, agendar serviços, registrar sugestões ou reclamações:
Ligue para: • Capitais e Regiões Metropolitanas: 4004.0014
• Outras Regiões: 0800 970 0999
Ou acesse o site www.consul.com.br
24 h por dia, 7 dias por semana, 365 dias por ano.

ÚNICO AUTORIZADO CONSUL:
- Mais de 3000 técnicos treinados
- Peças Originais
- Atendimento em horário comercial