UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS TRINDADE
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Alysson José Mendes Borba

Estação Meteorológica Inteligente

Florianópolis
2022

Alysson José Mendes Borba

Estação Meteorológica Inteligente

Relatório submetido à disciplina Programação de Sistemas Embarcados da Universidade Federal de Santa Catarina como requisito parcial à obtenção de grau referente ao semestre 2022.2.
Orientador: Prof. Eduardo Augusto Bezerra, Dr.

Florianópolis
2022

RESUMO

O presente relatório detalha a modelagem e desenvolvimento do projeto aplicado da Estação Meteorológica. O objetivo principal do projeto é otimizar o consumo de energia da estação utilizando técnicas de aprendizado de máquina para aprimorar o tráfego de dados oriundos da estação. São apresentados três *softwares* para diferentes plataformas (computador, *smartphone* e software embarcado), todos utilizando a linguagem C++. O microcontrolador utilizado é o Raspberry Pi, aliado a um sensor de temperatura LM35.

Palavras-chave: C++, estação meteorológica, microcontrolador, aprendizado de máquina, temperatura.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

É bastante comum que sistemas de monitoramento enviem dados continuamente, independentemente do valor a ser enviado. Em alguns casos, entretanto, não é necessário que esse envio seja feito caso os valores sejam redundantes, uma vez que isso pode ocasionar um uso excessivo de memória bem como diminuição na autonomia do sistema, já que muita energia é consumida para o envio dos dados.

Nesse contexto, técnicas de aprendizado de máquina (*machine learning*) podem ser implementadas visando aperfeiçoar tais sistemas, uma vez que, com base em um número de medições, é possível gerar um modelo que preveja o próximo dado. Dessa forma, é possível comparar uma novo dado com o modelo previsto, e, apenas se o novo valor estiver fora de uma margem de erro, o dado é enviado para um servidor.

A aplicação proposta nesse trabalho é uma estação meteorológica que coletará dados de temperatura. Em seguida, será feita uma comparação entre o dado lido e o previsto, e apenas se o novo valor lido estiver divergente e fora da margem de erro, o novo dado será enviado para o servidor.

O presente relatório visa documentar as etapas de modelagem e implementação do código da estação meteorológica de acordo com as especificações solicitadas.

1.1 OBJETIVOS

Este trabalho tem como objetivo geral a implementação de um projeto de estação meteorológica, utilizando técnicas de aprendizado de máquina e programação em C++, visando o aprimoramento da autonomia de bateria.

1.1.1 Objetivos específicos

- Exercitar a modelagem orientada a objetos visando implementação em C++;
- Entender os desafios do projeto integrado de *software/hardware* para sistemas embarcados;
- Praticar o fluxo completo de projeto de sistemas em C++ para sistemas embarcados.

2 DESENVOLVIMENTO

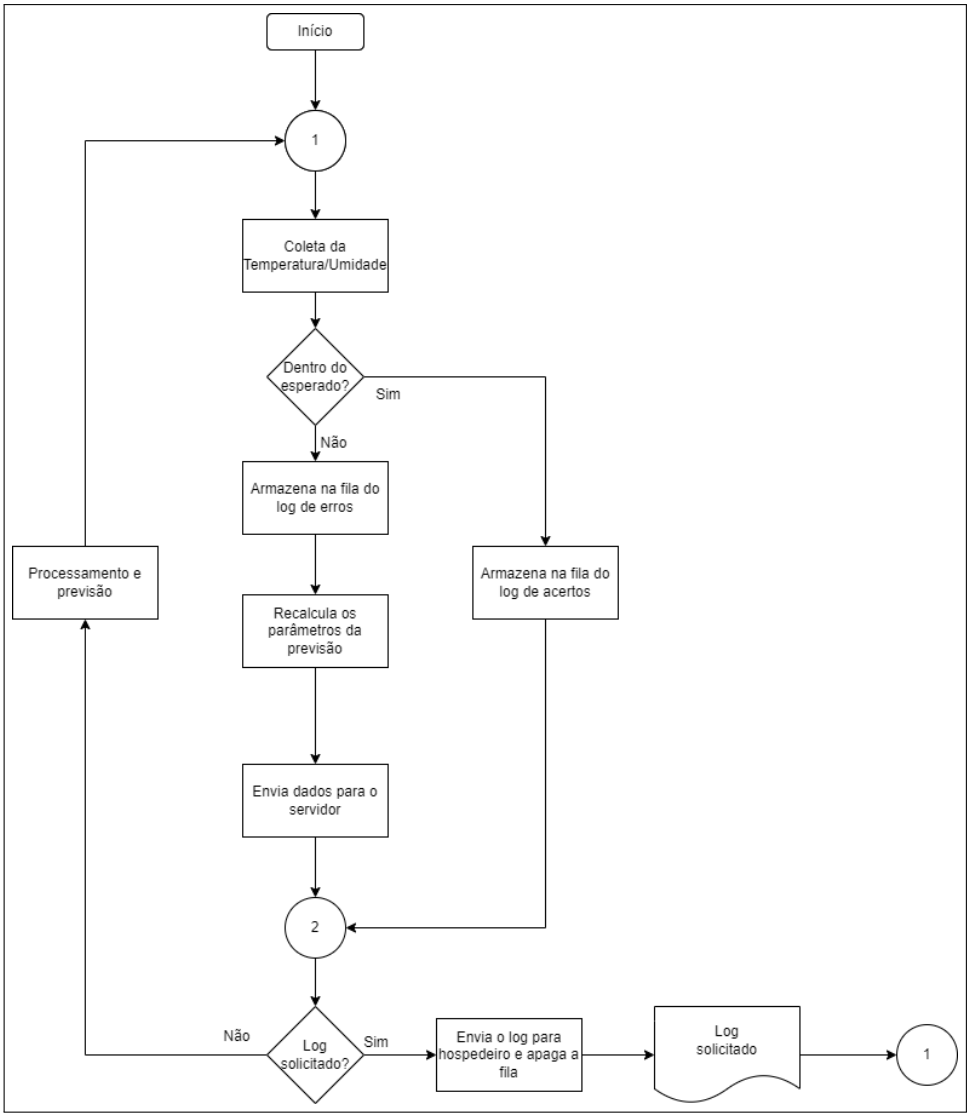
O projeto é dividido em três diferentes *softwares*, sendo um para o microcontrolador, um para o computador e, por fim, um para o *smartphone*.

2.1 SOFTWARE EMBARCADO

O *software* do microcontrolador é responsável por fazer a leitura dos sensores, armazenar os *logs* de certos e errados, enviar para o servidor as ocorrências fora da margem padrão e enviar todos os *logs* para o *software* hospedeiro (computador e *smartphone*) quando solicitado.

Essa etapa do projeto segue o fluxograma apresentado na Figura 1.

Figura 1 – Fluxograma do *Software* Embarcado.



Fonte: Elaborado pelo autor (2022).

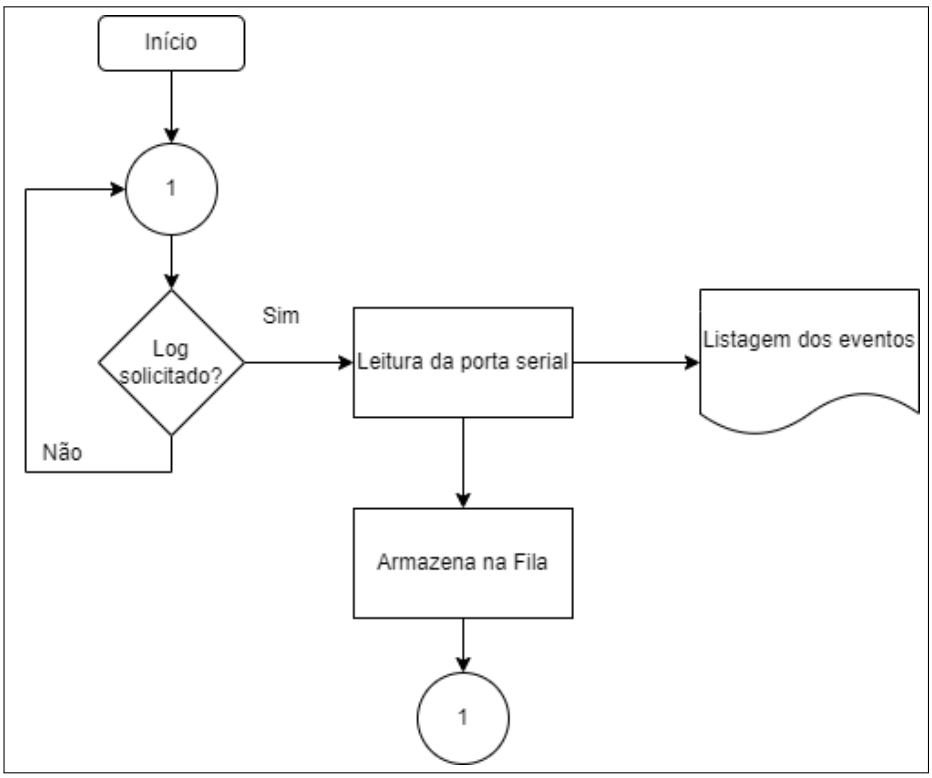
Inicialmente, é realizada a coleta da grandeza a ser medida, que nesse caso é a temperatura. Através dos métodos de aprendizado de máquina do *framework* TinyML, será realizado o treinamento do modelo para identificar os padrões e prever os próximos valores de temperatura. Caso a nova temperatura coletada esteja dentro do esperado, então esse valor é armazenado em uma fila de "acertos". Caso contrário, o valor é armazenado na fila de erros, os valores dos parâmetros de previsão são recalculados e, em seguida, enviados ao servidor. O programa fica na espera da solicitação do *log* por parte dos hospedeiros. Quando o *log* for solicitado, então os dados são enviados para o hospedeiro por meio da porta serial.

O *software* do sistema embarcado será melhor explorado nas próximas etapas do projeto.

2.2 *SOFTWARE* DO COMPUTADOR

O *software* do computador é responsável por listar todos os eventos ocorridos em um determinado intervalo de datas. Para isso, é utilizado a comunicação com a porta serial (com cabo) para recebimento das informações. O fluxo do código é mostrado na Figura 2.

Figura 2 – Fluxograma do *Software* do Computador.

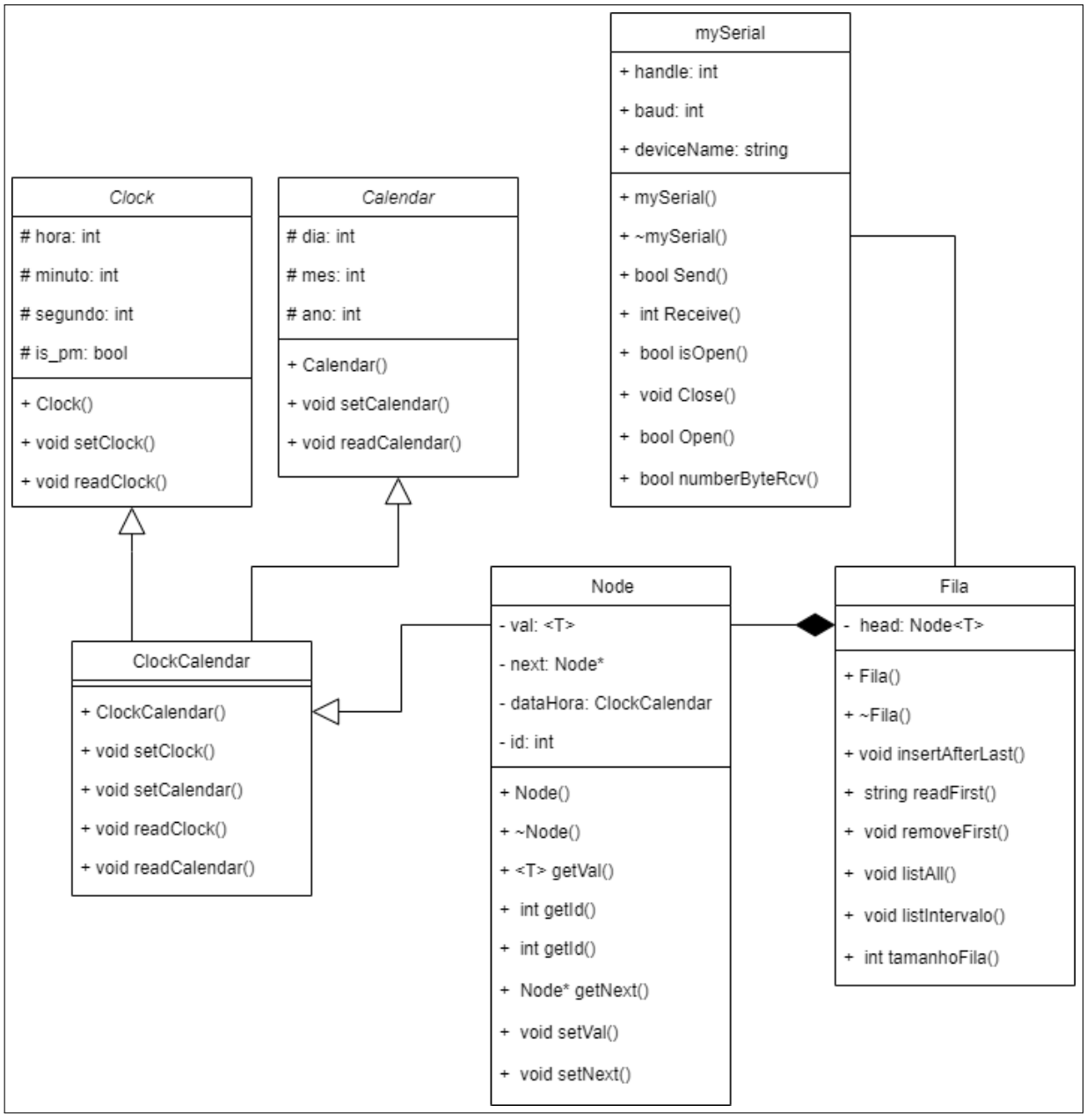


Fonte: Elaborado pelo autor (2022).

2.2.1 Diagrama de Classes

O digrama de classes, mostrado na Figura 3, explicita todas as classes, atributos e métodos utilizados para o desenvolvimento do *software* e é explicado a seguir.

Figura 3 – Diagrama de classes do projeto.



Fonte: Elaborado pelos autores (2022).

As classes *Clock* e *Calendar*, que já haviam sido criadas em aulas anteriores, bem como a classe filha ClockCalendar, que herda ambas, foram atualizadas para implementar o armazenamento da hora e data atuais, tendo em vista que as classes criadas anteriormente apenas inicializavam um objeto com data e hora fixas. Tal funcionalidade foi alcançada por meio da utilização da biblioteca padrão <ctime>. As classes

Clock e *Calendar* foram implementadas como classes abstratas, já que as funções de *set* foram declaradas como funções virtual pura, forçando a sua implementação na na classe filha *Clock Calendar*.

A funcionalidade de Fila, para armazenamento dos logs solicitados, foi realizado utilizando as classes Fila e Node. Um "nodo"é composto por um ID, referente à identificação do sensor, um objeto ClockCalendar, para definir o momento da leitura de um novo dado, o valor lido e um ponteiro para o próximo nodo da fila. O atributo valor, assim como as classes Node e Fila, foi declarado utilizando o modelo *template*, de forma que o código pode ser utilizado para armazenar diferentes tipos de dados. No esquema proposto, o tipo utilizado é o *float*, por se tratar de valores de temperatura.

A classe Fila foi construída de forma a possibilitar a implementação das funcionalidades de uma fila, com métodos para inserir depois do último, ler o primeiro, remover o primeiro, listar todos os elementos e listar elementos de acordo com um intervalo de datas, além de um método para retornar o tamanho da fila.

Por fim, a classe mySerial foi utilizada visando implementar a comunicação com a porta serial, tanto para transmitir como para receber os dados. Para isso, são definidos alguns atributos como nome da porta serial (deviceName), e a velocidade de comunicação (baud).

2.3 PLATAFORMA DE DESENVOLVIMENTO

A implementação do sistema será realizada utilizando a plataforma Raspberry Pi. A placa possui entradas micro USB (para alimentação e transferência de dados e áudio) e HDMI (para a transferência de vídeo), além de interfaces para redes sem fio, como Bluetooth e Wi-Fi. Essas características de facilidade de interfaceamento com os dados fazem da Raspberry Pi uma alternativa compacta e de baixo custo para o projeto proposto. Existem diversas versões do microcomputador.

A Raspberry Pi roda o sistema Linux e, por isso, a programação, que pode ser realizada em diversas linguagens, é semelhante à estrutura da programação para PC. Nesse projeto, será utilizada a linguagem C++, visando pôr em prática os conceitos aprendidos em sala de aula.

O microcomputador Raspberry Pi B+ possui 40 pinos, entre eles pinos de GPIO (*General Purpose Input/Output*, pinos de alimentação, e comunicação UART e I2C. Dessa forma, a Raspberry se torna um microcontrolador robusto, com capacidade computacional considerável e de baixo custo.

2.3.1 Sensor de Temperatura

Para coleta dos dados será utilizado o sensor de temperatura LM75, que se comunica através do protocolo I2C, possibilitando a sua aplicação com a Raspberry Pi.

Inicialmente, serão coletados dados visando o aprendizado de máquina, de forma a prever o próximo valor de temperatura.

2.4 *SOFTWARE* DO *SMARTPHONE*

O *software* do *smartphone* realizará a mesma tarefa do *software* do computador, isto é, receber e armazenar os logs quando solicitado. Para isso, será utilizada a plataforma Android Studio.

Para programar para Android em C++, no Android Studio, é necessário instalar o Android Native Development Kit (NDK), que é um conjunto de ferramentas que permite usar código C e C++, além de oferecer bibliotecas da plataforma para gerenciar atividades nativas e acessar componentes de dispositivos físicos, como sensores e entrada por toque.

Idealmente, seria realizada a comunicação por meio de uma interface WiFi ou Bluetooth. No entanto, como essa etapa não será implementada, serão utilizados dados sintéticos para representar as informações obtidas do controlador proposto. Nesse projeto, o software será construindo simulando uma conexão Bluetooth.

Mais detalhes sobre o *software* para celular serão apresentados nas etapas posteriores do projeto.

3 CONCLUSÃO

Para a primeira entrega, foram desenvolvidos o *software* de recepção e um esboço do *software* que irá no sistema embarcado. Foi utilizada a classe Fila para armazenamento dos logs tanto na transmissão como na recepção. Cada Nodo da Fila, possui um código identificador, as informações de data e hora e o valor de temperatura lido. A comunicação é feita com a classe mySerial, que permite a leitura e escrita na porta serial.

As demais funcionalidades serão apresentadas com o decorrer das entregas. O projeto pode ser acompanho por meio do repositório

https://github.com/alyssonmendes/estacao_meteorologica/.