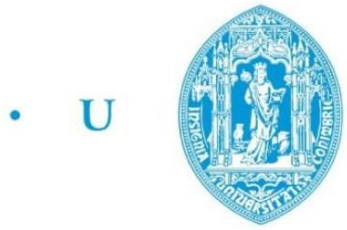


FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

PROJETO 2ª Meta

SISTEMAS OPERATIVOS

LICENCIATURA EM ENGENHARIA INFORMÁTICA

2022/2023

Francisco Henrique Baltazar Querido – 2021221158

INDICE:

1- Introdução:

O objetivo deste trabalho é desenvolver uma simulação simplificada de um ambiente de Internet das Coisas, onde diversos sensores enviam informações (leituras) para um ponto centralizado. Esse ponto armazena os dados, gera estatísticas e aciona alertas quando determinadas condições são atingidas.

Neste relatório será apresentada a arquitetura e a implementação deste ambiente simulado, bem como os desafios encontrados durante o desenvolvimento. O trabalho visa proporcionar uma compreensão mais aprofundada dos princípios e desafios envolvidos na criação de sistemas de IoT, contribuindo para o avanço nessa área em constante evolução.

2- Funcionamento

2.1- Ficheiros

Para a implementação deste projeto, foram usados:

- 3 ficheiros fonte (‘User_console.c’, ‘System_manager.c’ e ‘Sensor.c’);

- 3 ficheiros header (‘home_iot.h’, ‘console.h’, ‘sensor.h’);

- 1 ficheiro Makefile;

- 1 ficheiro de configurações (‘config.txt’).

2.2 – Recursos

Para o funcionamento deste projeto são usados os seguintes recursos:

-FIFO SENSOR_PIPE - Permite que os processos Sensor enviem dados ao processo System Manager;

-FIFO CONSOLE_PIPE - Usado para enviar comandos efetuados pelo User Console;

-SHM - Zona de memória partilhada acedida pelos processos System Manager, Worker e Alerts Watcher;

-Unnamed pipes - Permitem a comunicação entre o processo System Manager e cada um dos processos Worker;

-Message Queue - Permite o envio das mensagens de alerta a partir do Alerts Watcher para os User Consoles, e o envio das respostas aos pedidos dos User Consoles.

-Internal queue – Permite o envio dos comandos lidos pelos pipes para serem processados posteriormente pela dispatcher.

2.3 – System Manager

O programa inicia quando o ficheiro 'System_manager.c' é compilado e executado. Ao inicializar lê os dados do ficheiro de configurações e cria (e abre se for o caso) todos os recursos necessários para o seu funcionamento, cria também os processos 'Worker', o processo 'Alerts Whatcher' e as threads 'console_thread', 'sensor_thread' e 'dispatcher_thread'.

A 'console_thread' lê os dados do FIFO "CONSOLE_PIPE" e coloca-os na internal queue para serem processados pela 'dispatcher_thread'.

De forma semelhante, a 'sensor_thread' lê os dados do FIFO "SENSOR_PIPE" e coloca-os também na internal queue para serem processados pela 'dispatcher_thread'.

A 'dispatcher_thread' lê os comandos presentes na internal queue, verifica que Worker está disponível e envia o comando lido para unnamed pipe do mesmo. Esta thread dá prioridade aos comandos vindos da 'console_thread'.

Os processos 'Worker' recebem o comando enviado pela dispatcher do unnamed pipe, analisa-o para verificar se é proveniente do sensor ou da consola e processa o mesmo consoante a sua origem. Se for do sensor, cria um novo ou atualiza as estatísticas do mesmo caso já exista em memória e sinaliza ao 'alerts watcher' que houve uma modificação nos sensores para que este a verifique. Se for da consola, executa a diretiva enviada pela mesma e devolve uma mensagem à consola que enviou o comando a informar sobre o sucesso da operação.

O processo 'Alerts Watcher' fica bloqueado à espera que um processo Worker sinalize a variável de condição para que este acorde e percorra os dados de todos os sensores para verificar se a ultima leitura realizada está fora dos limites estipulados pelos alertas existentes no sistema, caso esteja fora desses parâmetros envia uma mensagem à consola que criou o alerta a informar da infração.

Ao receber o sinal SIGINT o programa entra na função 'cleanup()' que liberta todos os recursos usados pelo sistema, encerra os processos e threads em execução e termina o programa.

2.4 – User Console

Ao compilar e executar o ficheiro 'User_console.c' o programa começa por abrir e inicializar os recursos necessários para a sua execução e a thread para ler da message queue. Posteriormente apresenta um menu interativo ao utilizador para que escolha a opção a realizar e espera o seu input. Assim que o recebe, envia para o processo começado pelo "System_manager.c" através do FIFO "CONSOLE_PIPE".

A thread está constantemente a tentar ler da message queue, onde recebe dados do System Manager enviados pelos processos 'Worker' e 'Alerts Watcher' e assim que recebe uma mensagem imprime-a mesma no ecrã.

Ao receber o sinal SIGINT o programa entra na função 'cleanup()' que liberta todos os recursos usados pelo sistema, encerra a thread em execução e termina o programa.

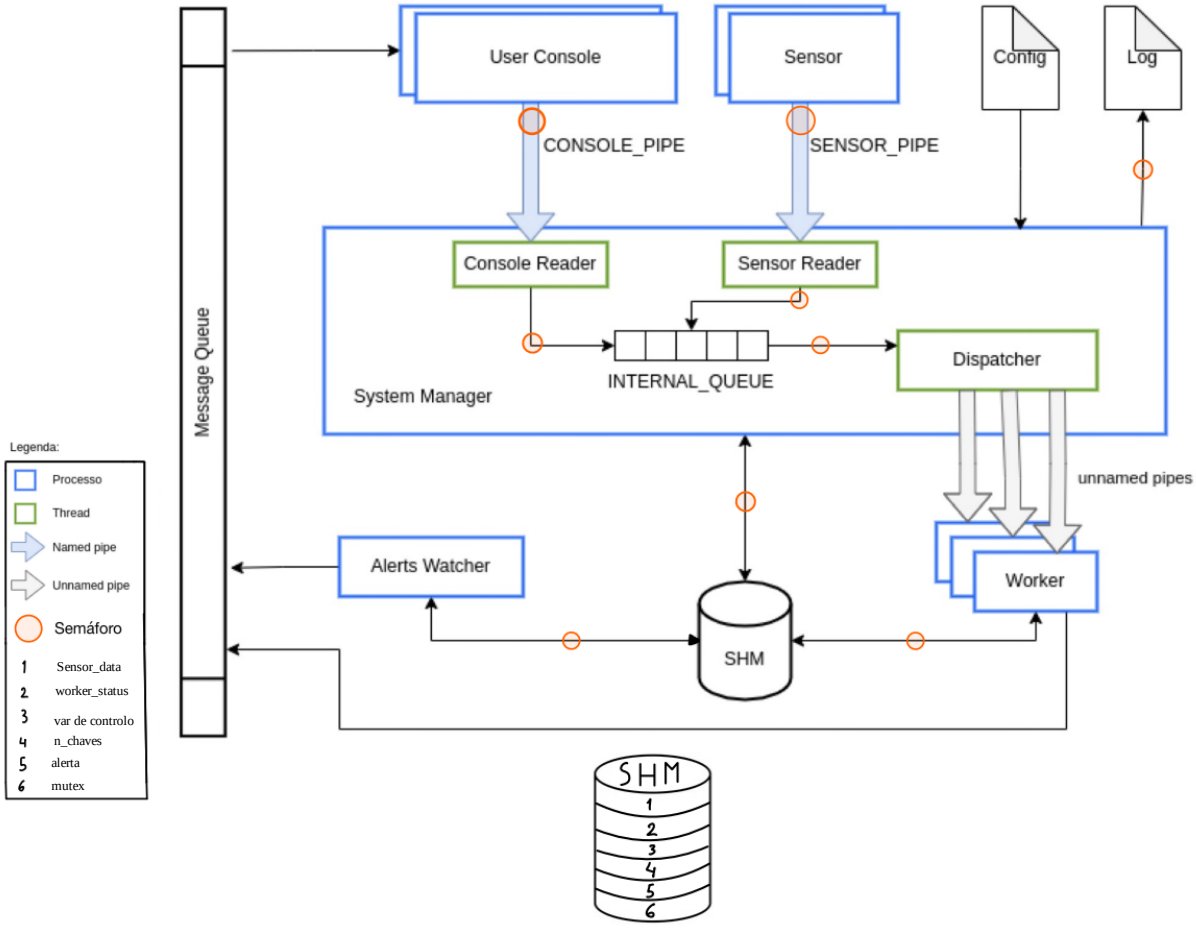
2.5 – Sensor

Ao compilar e executar o ficheiro 'Sensor.c' o programa começa por abrir e inicializar os recursos necessários para a sua execução e cria uma variável do tipo 'Sensor' com os dados introduzidos pelo utilizador na linha comandos aquando da execução do programa. Posteriormente entra numa função onde gera a última leitura que consiste num número aleatório dentro dos parâmetros definidos e envia a cada intervalo de tempo, também estipulado pelo utilizador, através do FIFO "SENSOR_PIPE" o id, a chave e a leitura do sensor em questão.

Ao receber o sinal SIGTSTP imprime no ecrã o número de mensagens geradas e enviadas até o momento para o User Console.

Ao receber o sinal SIGINT o programa entra na função 'cleanup()' que liberta todos os recursos usados pelo sistema e encerra o programa.

3- Esquema da Arquitetura do Programa



4- Tempo Total Despendido

Aproximadamente 60h