Sistemas Operativos 2017/2018

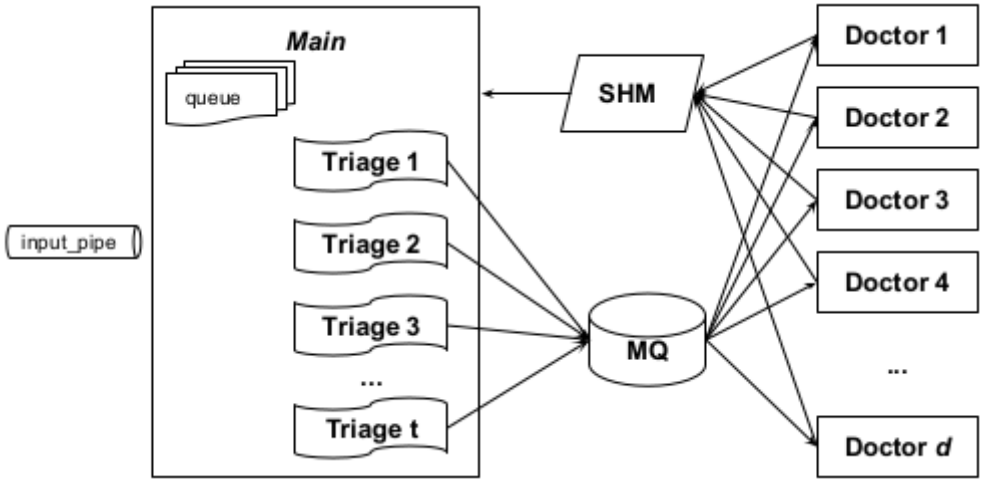
Projeto

Gestão de Urgências

Introdução

Foi nos pedido desenvolver um simulador das urgências de um hospital capaz de representar os processos de triagem e de atendimento que decorrem num hospital. Esse simulador foi desenvolvido na linguagem de programação C sendo que tinha como objetivo explorar mecanismos de gestão de processos, threads, comunicação e sincronização em Linux.

Arquitetura



- ## Estruturas

O nosso programa é constituído por 5 estruturas:

- ➔ **Paciente –** Tem como atributos o nome, a prioridade obtida, o nº de chegada, o tempo do processo da triagem, o tempo do processo do atendimento e também as datas de chegada ao hospital e de saída da triagem.
- ➔ **Message –** Tem como atributos o paciente e a prioridade.
- ➔ **Q_type –** Estrutura usada como fila de espera.
- ➔ **Node-type –** Estrutura usada como nó da fila de espera. Tem como atributo o paciente.

➔ **Stats –** Estrutura que vai ser mapeada em memória partilhada. Guarda todas as informações partilhadas entre processos e threads. Tem como atributos 4 mutexes (memória partilhada, fila de espera, fila de mensagens e ficheiro log), número de processos e array de processos, array de threads e respetivos ids, dados relativos às configurações do programa (nº de threads, processos, tamanho dos turnos e máximo da fila), dados relativos às estatísticas do programa (pacientes atendidos, pacientes triados, tempo de espera na fila para a triagem, tempo de espera na fila para o atendimento e tempo total de espera), entre outras variáveis auxiliares essenciais para o correto funcionamento do programa.

- # Funcionamento do Programa

Processo Principal - Começa por abrir o ficheiro de log. Depois cria o named pipe, a fila de espera, mapeia em memória partilhada e inicializa a estrutura Stats, lê as configurações do ficheiro config.txt e começa a criar as threads. Cria o nº de threads indicados no parâmetro TRIAGE do ficheiro config.txt e uma outra thread extra responsável por estar à escuta no named pipe. Depois de criadas, o processo principal começa a criar os processos doutores, esperando depois que eles terminem o turno para criar novos processos doutores. Quando recebe um sinal de SIGINT, o processo principal limpa todos os recursos e acaba o programa.

Thread Extra – Esta thread fica à espera que chegue um pedido através do named pipe. O pedido pode ser a chegada de um paciente individual, um grupo de pacientes, ou um comando especial que altera dinamicamente o nº de threads ativas.

Threads – As threads de triagem começam por retirar um paciente da fila de espera (se existir) , esperam o tempo corresponde ao processo de triagem, atribuem a prioridade ao paciente (valor recebido no pedido feito através do named pipe) e inserem como mensagem numa fila de mensagens, depois disso atualizam as estatísticas e voltam a repetir todo o processo até que sejam mandadas terminar.

Processos – Os processos doutores começam por retirar uma mensagem da fila de mensagens, esperam o tempo correspondente ao processo de atendimento, atualizam as estatísticas e voltam a repetir o processo até o turno terminar ou sejam mandados parar.

Tempo gasto no desenvolvimento do Projeto

1ª Meta – 8 horas

Meta final – 48 horas

Tempo Total – 56 horas

Trabalho realizado por:

Gonçalo Filipe Lucas Menino Rodrigues Pinto nº2014202176

João Filipe Mendes Castilho nº2014202807