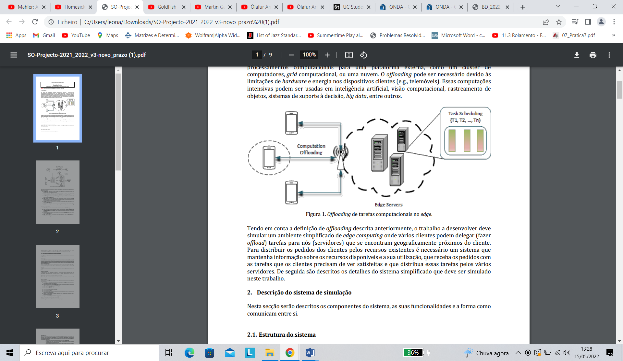
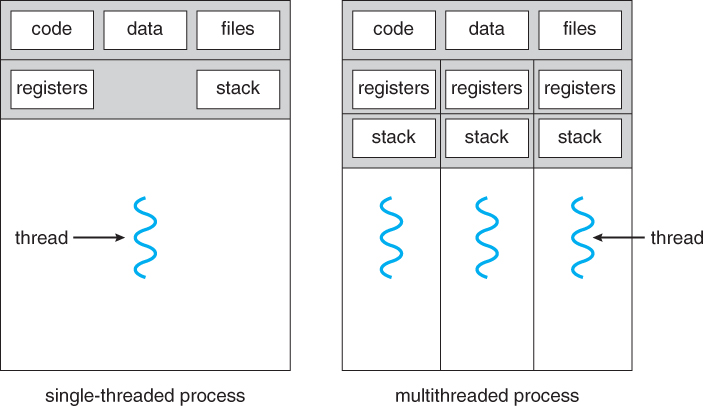
Uma imagem com texto

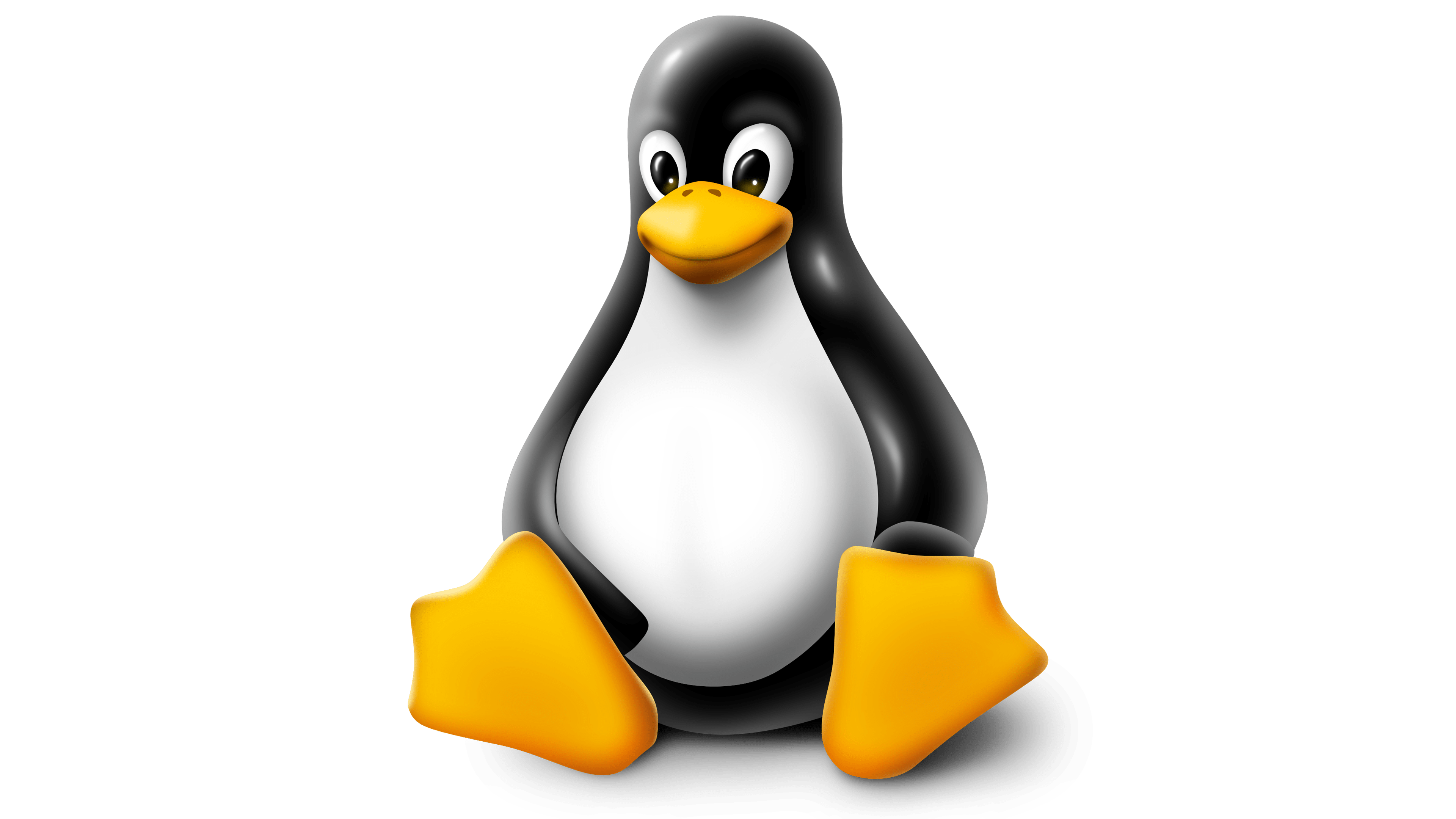
Descrição gerada automaticamente **FACULDADE DE CIÊNCIAS E TECNOLOGIA**

**DEPARTAMENTO DE ENGENHARIA INFORMÁTICA**

Sistemas operativos

Simulador para Offloading de Tarefas no Edge





Trabalho realizado por:

João Moura (2020235800)

Luís Neto (…)

Introdução:

O nosso projeto tem como objetivo a implementação de um simulador de offloading que através do uso de diferentes servidores (***edge servers***) que irão processar diferentes tarefas enviadas pelo processo ***mobile node.*** Para a implementação das diversas funcionalidades utilizamos essencialmente conhecimentos relativos a ***processos*** e a ***threads*** e ainda a ***mecanismos de sincronização*** (mutex,etc.).

Mobile Node:

Este processo irá enviar tarefas ao ***System manager*** através de um ***Named Pipe***, sendo executado com os parâmetros: número de pedidos a enviar, o intervalo entre pedidos, o número em milhares de instruções por pedido e o tempo de execução máximo por tarefa. Pelo ***Pipe*** mencionado anteriormente é enviado uma string com o índice do pedido, número de instruções em milhares e o tempo máximo para execução.

System Manager:

Processo principal (main) responsável pela criação e gestão dos restantes processos. Irá inicializar a memória partilhada, inicialização de variáveis e condições, mecanismos de sincronização, captura os sinais e apaga os recursos no final da execução do programa.

Task Manager:

O ***task manager*** é o processo responsável pela criação dos processos ***edge servers*** e das threads ***scheduler*** e ***dispatcher.*** Inicialmente abre um ***unnamed pipe*** para cada ***edge server*** e inicializa a fila de tarefas, de seguida cria as threads mencionadas anteriormente e os processos relativos a cada ***edge server.*** Entra então num ciclo ***while*** que será quebrado apenas quando a variável que se encontra em memória partilhada ***acabar\_prog*** for ativado e o programa terminar. Neste ciclo vão ser lidas as tarefas enviadas pelo ***Named Pipe*** que está em comunicação com o ***mobile node.*** De seguida caso a mensagem recebida não seja EXIT (que fará com que a variável de terminar o programa seja ativa) irá criar a nova tarefa e adicionar à fila de tarefas. Caso não haja espaço na fila a tarefa será apagada. Por última ativa a thread ***scheduler.***

Monitor:

Este processo ativa o modo “high performance”, ativa todos os ***vCPUs*** de todos os ***edge servers*** para melhorar a capacidade de processamento das tarefas. Este vai ativar uma flag que será lida pelo ***edge server*** caso a fila esteja 80% para ativar o modo “High Performance” e quando está inferior a 20% volta ao estado normal recorrendo à flag para comunicar.

Edge Server:

O Edge Server é responsável por iniciar as duas threads correspondentes a cada um dos ***vCPUs*** e de seguida definir o modo de performance de cada um, podendo ser “Stopped”,” Normal” ou “High Performance”. De seguida cada ***edge server*** irá ler do seu ***unnamed pipe*** específico o número de instruções a executar no vCPU disponível, guardando estas numa estrutura específica para cada tarefa.

Dispatcher:

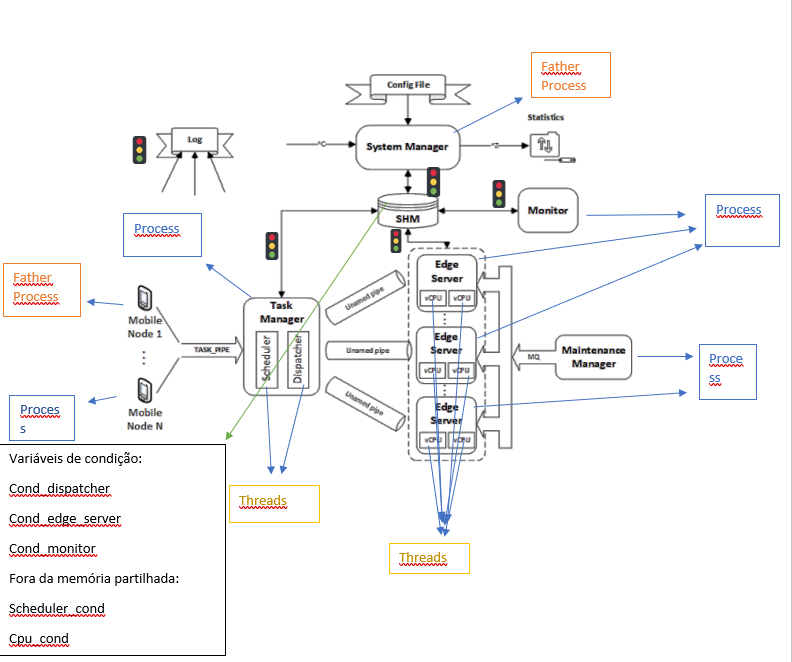
Esta thread criada pelo ***Task Manager*** irá escolher um dos vCPUs disponíveis (que não estejam em modo “Stopped”) para executar a tarefa caso esta esteja abaixo do tempo máximo para execução da tarefa senão apaga a tarefa. Caso supere o tempo máximo de espera para uma tarefa executar envia um sinal para ativar o monitor. Após a seleção do ***edge server*** e do ***vCPU*** envia a informação de cada tarefa pelo ***unnamed pipe*** correspondente ao ***edge server.***

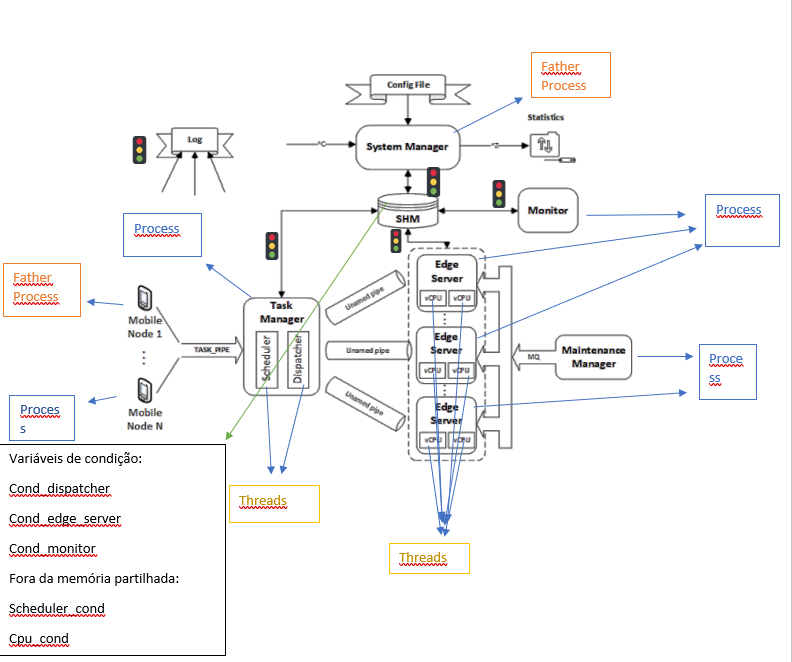
Scheduler:

O ***scheduler*** é uma thread criada pelo ***Task Manager*** responsável por organizar a fila de tarefas. Quando entra uma nova tarefa na fila o ***Task Manager*** envia um sinal para ativar o ***scheduler*** para reorganizar a fila por ordem crescente de tempo máximo de execução de tarefas. Em seguida envia um sinal para o ***dispatcher*** para continuar a enviar as tarefas.

Maintenance Manager:

Este processo é responsável pelo envio de mensagem e gerir a manutenção aleatória dos vários ***edge servers*** através do uso de uma message queue (incluída na biblioteca <sys/msg.h>). O processo só será responsável por receber uma mensagem que termina as threads iniciadas (msgtype-404) e iniciar as mesmas. Estas threads serão o ***maintenance\_reader*** que irá receber a mensagem relativa ao ínicio de uma manutenção de um servidor colocando esse servidor a “Stopped” e quando receber a mensagem para continuar irá recolocar em “Normal”. Também poderá receber enviar uma mensagem para terminar para processo de manutenção. A outra thread ***maintenance\_thread*** será responsávelpor enviar a mensagem de que o servidor irá entrar em manutenção e por receber a mensagem da outra thread a mensagem de entrar em manutenção colocando então o servidor escolhido em manutenção. Caso receba a mensagem “TERMINA” a thread morre. Após fazer a manutenção envia uma mensagem para o servidor reentrar em atividade.

Diagrama de Arquitetura:



Conclusão:

Este projeto permitiu consolidar os conhecimentos aprendidos nesta cadeira e solidificar a capacidade de trabalho em equipa.

Em termos de divisão de trabalho, foi relativamente equilibrado dado que ambos fizemos mais ou menos o mesmo esforço tornando a execução do projeto mais eficiente.