Instituto Politécnico de Viseu
Escola Superior de Tecnologia e Gestão de Viseu
Departamento de informática

Relatório do projeto de Sistemas Distribuídos

Licenciatura em Engenharia Informática

Dany Almeida nº 16393
Tiago Simões nº 15592
Daniel Braga nº 16623
Pedro Coelho nº 16312

Viseu, 2022

Instituto Politécnico de Viseu
Escola Superior de Tecnologia e Gestão de Viseu
Departamento de informática

Relatório do projeto de Sistemas Distribuídos
Licenciatura em Engenharia Informática
3º ano
1º semestre
Ano letivo 2021/2022

Dany Almeida nº 16393
Tiago Simões nº 15592
Daniel Braga nº 16623
Pedro Coelho nº 16312

Viseu, 2022

Índice

Índice de figuras

1. Introdução

O presente projeto, é elaborado no âmbito da disciplina Sistemas Distribuídos de Engenharia Informática do 3ºano 1º semestre, no Instituto Politécnico de Viseu, a decorrer no período de 19 de setembro de 2021 a 26 de fevereiro 2022.

A área de inteligência artificial exige uma grande disponibilidade de recursos computacionais. O CPU é o recurso mais importante nesta área, uma vez que os algoritmos são computacionalmente muito intensivos. Como tal, num sistema distribuído torna-se fundamental rentabilizar a utilização deste recurso nos vários elementos computacionais.

Com o desenvolvimento deste projeto pretende-se apresentar uma solução distribuída que permita distribuir os pedidos de computação pelos elementos disponíveis do sistema, de forma a garantir o equilíbrio da carga entre os mesmos.

Existem vários tipos de elementos no sistema. Os processadores são elementos que executam os algoritmos. Os carregadores são elementos que armazenam os dados a processar. Os cérebros são elementos que armazenam os modelos gerados. Os estabilizadores garantem a distribuição equilibrada da carga pelos processadores.

2. Arquitetura da solução em UML

2.1. UML

2.1.1. UML Balanceador

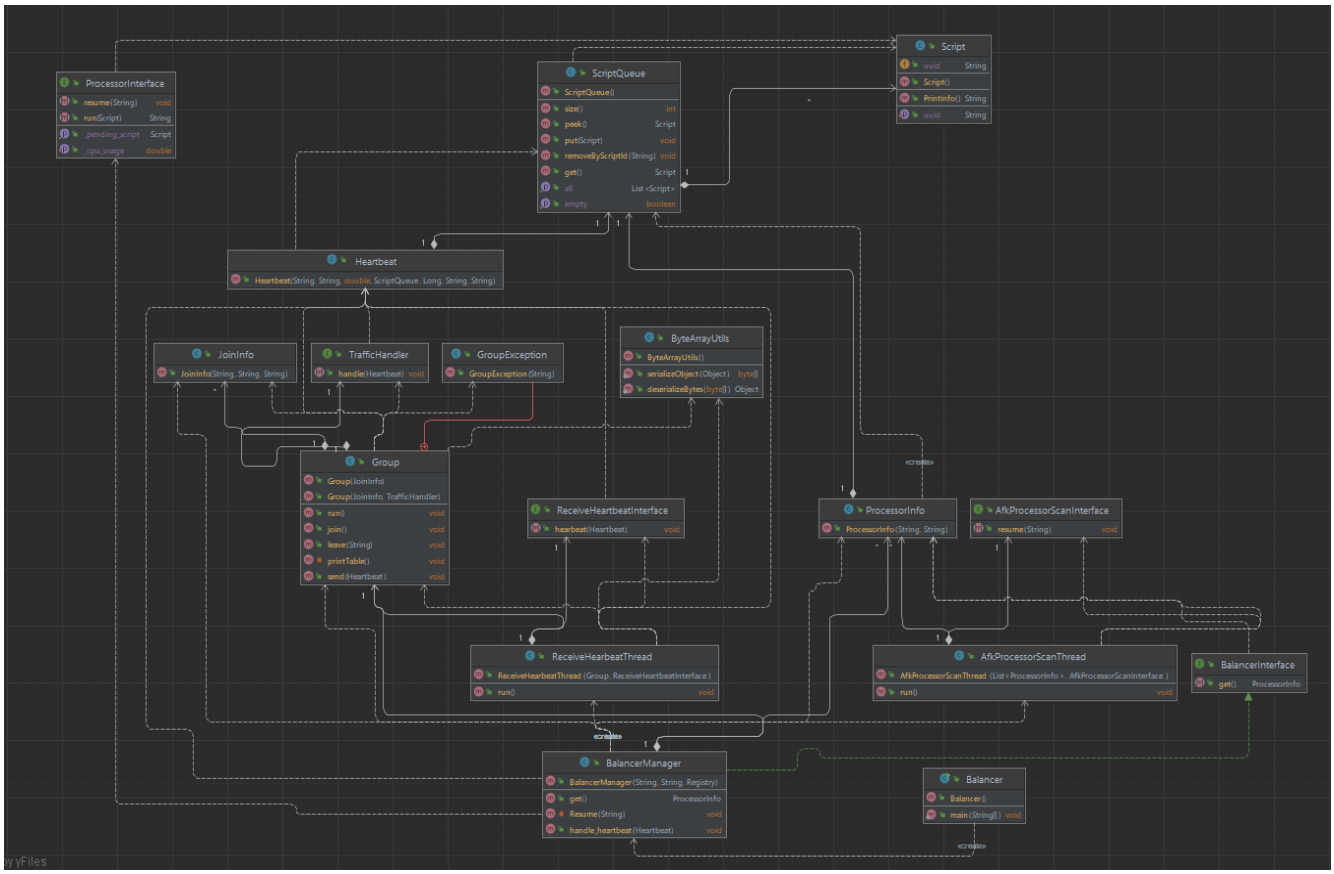

Figura 1 - UML Balanceador

2.1.2. UML Cérebro

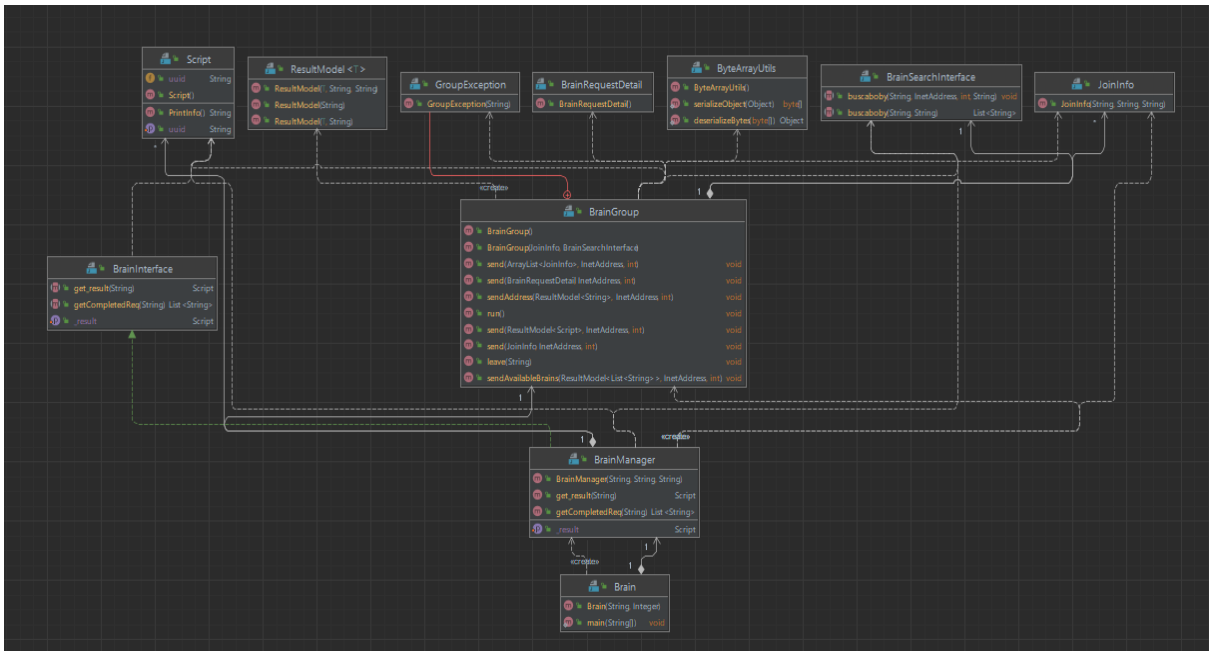

Figura 2 - UML Cérebro

2.1.3. UML Cliente

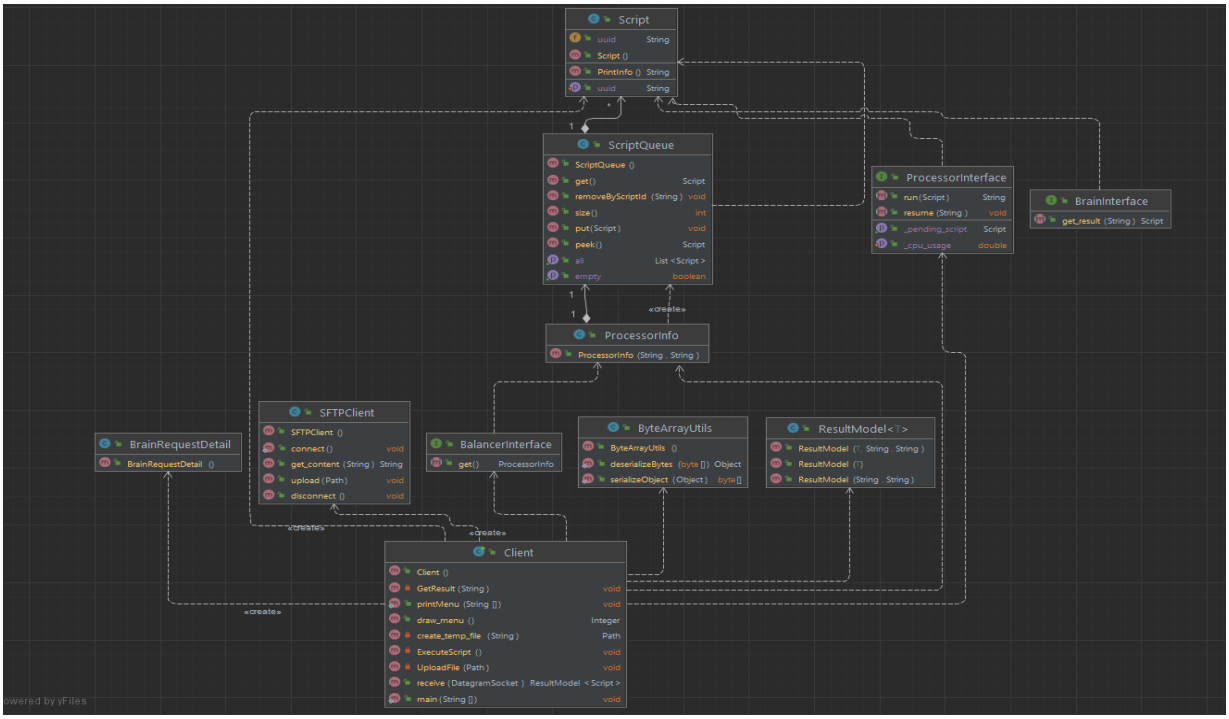

Figura 3 - UML Cliente

2.1.4. UML Processador

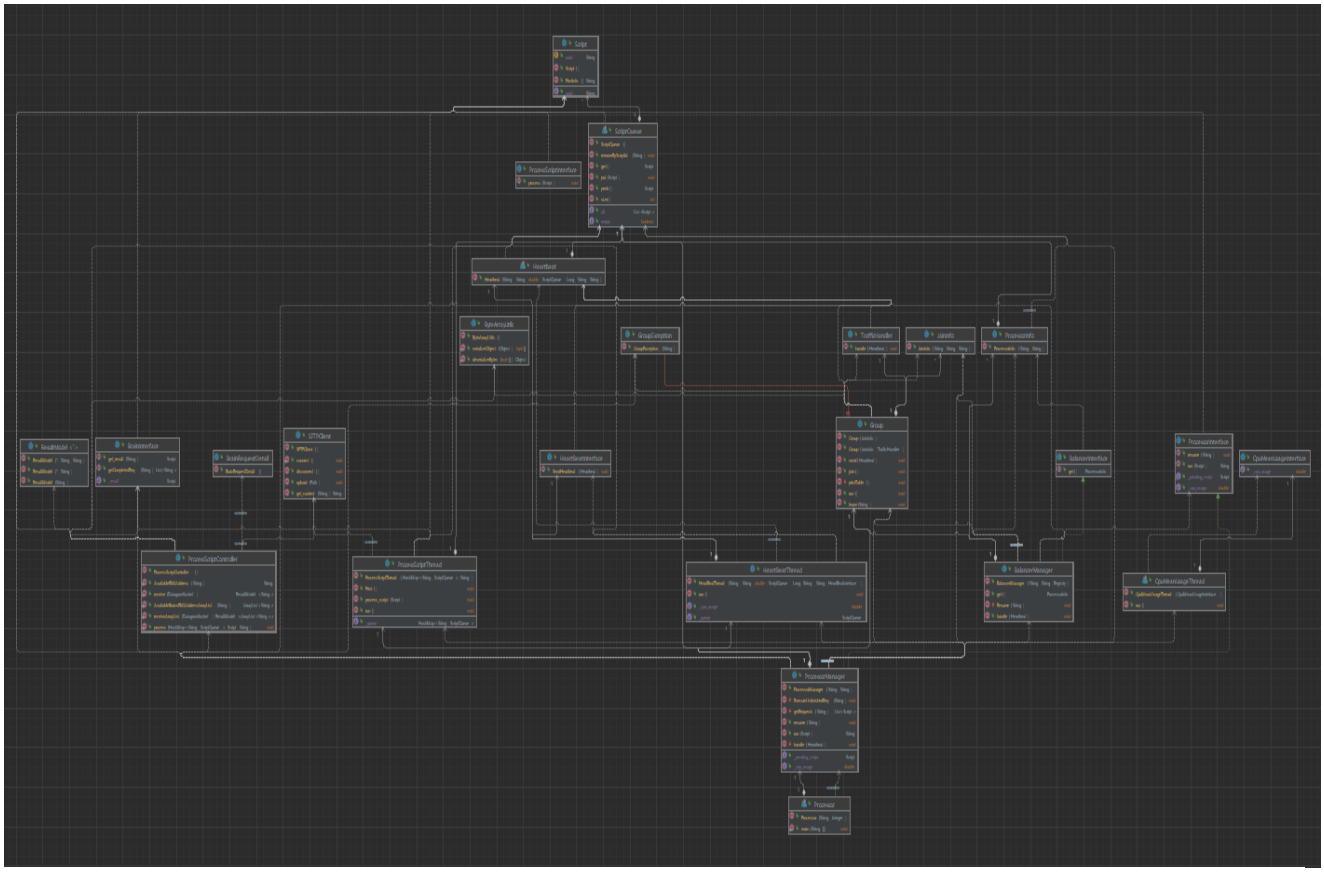

Figura 4 - UML Processador

3. Implementação

Neste trabalho foram desenvolvidos 4 projetos, Cliente, Balanceador, Processador e Cérebro. Foi implementado com especial foque os grupos Multicast para comunicações entre balanceadores, processadores, cérebros e clientes.

No balanceador foi implementado um serviço RMI para o cliente consultar e receber o endereço do processador com recursos mais livres, estes recursos são enviados posteriormente por *hearbeats* dos processadores no grupo Multicast também implementado para o balanceador que se encontra no mesmo grupo. Assim que um processador deixe de enviar *hearbeats* após 30 segundos o balanceador vai então remover o mesmo da sua lista de processadores ativos e vai pedir a um processador com menos recursos que processe os scripts do processador que se desligou, antes de processar os scripts o processador irá aderir ao grupo Multicast para receber todos os endereços dos cérebros ativos e irá 1 a 1 saber quais os pedidos já processados e descartá-los e apenas processar aqueles que não se encontram nos cérebros.

Se o balanceador se desligar ou deixar de estar operacional foi adicionado um *hook* de *shutdown* que envia um *hearbeat* do tipo *failed* para um processador se tornar o balanceador, após isto o mesmo remove o objeto atual do balanceador presente no *Registry* e retira-se o balanceador do grupo Multicast.

No processador foi implementada uma *thread* para envio de *hearbeat* periódicos onde este tempo periódicos é totalmente gerado aleatoriamente em cada processador, uma *thread* para calcular a média de recursos usados na CPU, uma *thread* para se juntar ao grupo Multicast e um shutdown Hook para remover o processador do grupo multicast quando o mesmo se desliga, no método *handle* (localizado na classe ProcessorManager do sub-módulo do Processador) fica responsável por receber os *hearbeats* dos processadores com os seus scripts em lista de processamento, ou do balanceador em caso de falha em que o tipo de *hearbeat* é "*failed*".

No cérebro foi implementado um grupo multicast onde o mesmo pode receber pedidos para que seja retornado os endereços de todos os cérebros ativos e também um resultado de um script efetuado pelo cliente.

4. Conclusão

Com a conclusão deste trabalho aprendemos, a utilizar vários tipos de comunicação entre processos, entre eles o mais fascinante o Multicast, onde os processos se encontram todos num grupo onde podem comunicar todos uns com os outros.

Foi possível verificar que o CPU é um componente bastante importante e como os algoritmos são computacionalmente muito intensivos devemos geri-lo da melhor forma para que tiremos o máximo partido dele e assim evitarmos sobrecargas.

Acabamos por encontrar dificuldades para receber informação dos scripts guardados em todos os cérebros em uma apenas chamada, ultrapassando essa dificuldade fazendo um “dicionário” no grupo Multicast com todos os endereços RMI ativos dos cérebros permitindo-nos consultar todos os cérebros.

A elaboração deste projeto, permite tirar elações de desenvolvimento tanto a nível pessoal, como académico e profissional.