ISEL
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Trabalho Prático Avaliação Final

Computação Distribuída

Grupo 04:

Duarte Valente | A47657

João Valido | A51090

David Monteiro | A52120

Docente:

Eng. Luís Assunção

Curso: MEIM

2023

Índice

Índice de ilustrações

Introdução

A computação distribuída é uma área crucial na ciência da computação, permitindo a criação de sistemas escaláveis e robustos. Este trabalho prático tem como objetivo a implementação de um sistema distribuído para uma cadeia de supermercados, explorando alguns exemplos de interação. O sistema é configurado em máquinas virtuais na Google Cloud Platform, utilizando o Gluster File System para garantir a replicação de arquivos entre as VMs. Este capítulo apresenta uma visão geral do projeto, destacando o contexto, o propósito e os objetivos gerais.

a) Contextualização

Os supermercados enfrentam diversos desafios na gestão de grandes volumes de dados de vendas. Esses desafios incluem a necessidade de lidar com a escalabilidade operacional devido ao aumento constante do número de transações, a diversidade de dados provenientes de diferentes fontes, a demanda por informações em tempo real para tomada de decisões imediatas, a integração de sistemas diversos para criar uma visão unificada. Abordar esses desafios requer a implementação de sistemas distribuídos eficientes, capazes de processar e analisar dados de maneira escalável e em tempo real.

Uma abordagem distribuída aborda esses desafios ao permitir o processamento paralelo de grandes volumes de transações, garantir a atualização em tempo real das informações e adaptar-se dinamicamente a variações na demanda. Em resumo, a solução distribuída não apenas aborda os desafios presentes, mas também oferece flexibilidade para enfrentar futuras demandas e evoluções tecnológicas.

O sistema distribuído proposto no trabalho prático visa enfrentar esses desafios específicos no contexto de uma cadeia de supermercados.

b) Objetivo

O projeto visa desenvolver um sistema distribuído que simula o funcionamento de uma cadeia de supermercados, enfrentando desafios específicos, como o processamento eficiente de grandes volumes de dados de vendas. A arquitetura adotada, apresentada na Figura 1, utiliza máquinas virtuais na Google Cloud Platform e um sistema de ficheiros distribuídos para garantir escalabilidade e replicação eficiente de dados. Os objetivos principais incluem a implementação de uma aplicação de ponto de venda (User App), a utilização do paradigma Publish/Subscribe (RabbitMQ Docker) para distribuição de mensagens, a integração de comunicação em grupo e multicast para coordenar operações, e a demonstração da eficiência de um algoritmo de eleição para liderar operações específicas. Esses elementos combinam-se para criar uma solução prática, capaz de abordar as tarefas operacionais de uma cadeia de supermercados.

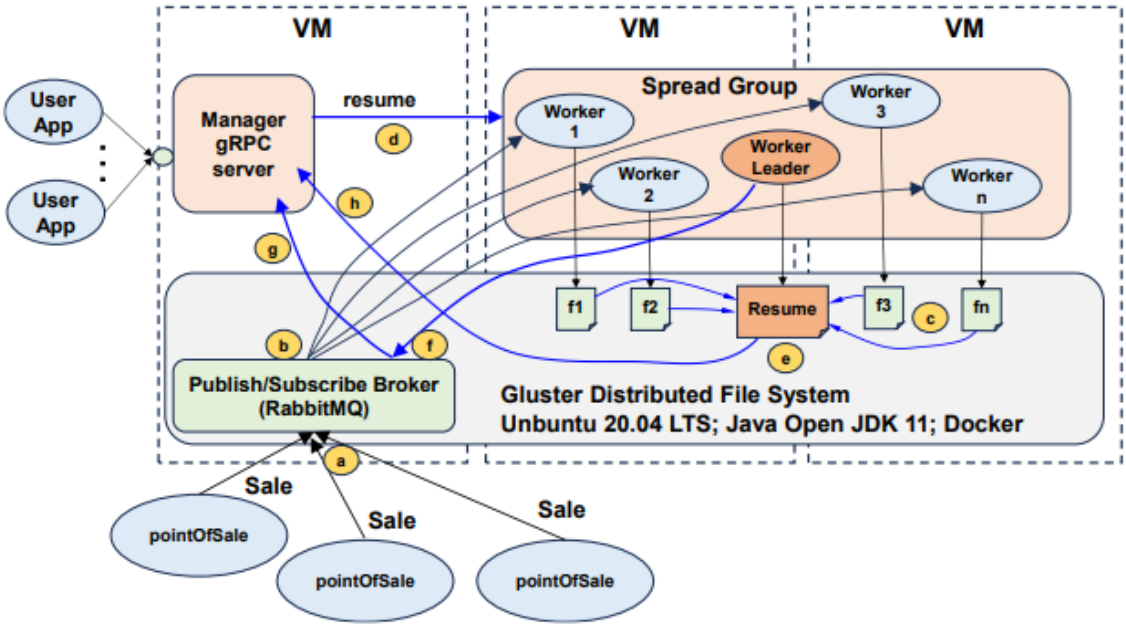

Figura 1 - Diagrama geral do sistema

Desenvolvimento

a) Arquitetura do Sistema

A arquitetura do sistema desempenha um papel central na eficiência e no funcionamento coerente do sistema. De seguida, são explorados os principais elementos que compõem essa arquitetura, destacando a interconexão entre eles.

1. Aplicação Point of Sale

A aplicação PointOfSale desempenha um papel central na simulação das operações de caixas de supermercado. Tendo como função principal gerar mensagens de vendas que contêm informações detalhadas sobre cada transação realizada nas caixas.

Atua como fonte primária de dados, gerando mensagens de vendas em resposta a cada transação realizada em um ponto de venda. As mensagens são enviadas para o Broker Publish/Subscribe para distribuição eficiente para outros componentes do sistema. A comunicação assíncrona com o Broker permite que a aplicação PointOfSale continue suas operações sem depender diretamente do processamento subsequente das mensagens.

Como opções de envio, optamos por simplificar esta funcionalidade ao permitir enviar 10 diferentes mensagens de produtos do tipo “Alimentar”, ou de enviar 10 diferentes mensagens do tipo “Casa”.

2. Broker Publish/Subscribe (Docker Container - RabbitMQ)

O Broker, implementado como um Docker container, utiliza o RabbitMQ como a plataforma de comunicação que suporta o sistema Publish/Subscribe. O RabbitMQ atua como um intermediário eficiente na distribuição assíncrona de mensagens, fornecendo um mecanismo robusto para a comunicação entre os diferentes componentes do sistema distribuído. Recebe mensagens de vendas geradas pela aplicação PointOfSale. O RabbitMQ, distribui essas mensagens para os Workers e outros elementos. O desacoplamento entre produtores e consumidores é facilitado pela lógica de troca de mensagens do RabbitMQ.

Para conseguir implementar o sistema pedido, foram criados dois exchanges, que são pontos de entrada para mensagens no RabbitMQ. As mensagens enviadas pela aplicação PointOfSale são encaminhadas para os Workers através dos exchanges. ExgResumo e ExgSales são os nomes dos exchanges criados, como demonstrado na Figura 2, que realizam o Worker Leader e o Manager gRPC Server e ainda entre o PointOfSale e os Workers respetivamente.

Virtual host	△ Name	Type	Features	Message rate in	Message rate out
/	ExgResumo	direct	D	0.00/s	0.00/s
/	ExgSales	topic	D	0.00/s	0.00/s

Figura 2 - Exchanges no RabbitMQ

Foram criadas ainda 3 Queues distintas, que são os locais onde as mensagens são armazenadas temporariamente antes de serem consumidas pelos destinatários finais. Cada tipo de mensagem é encaminhado para uma queue específica, garantindo a segregação e a ordenação adequada das mensagens. Para fazer a ligação com os exchanges, fazemos o binding das queues, que é o processo de conectar uma queue a um exchange. Cada Worker cria uma queue e vincula a mesma a um exchange específico. Isso assegura que cada tipo de mensagem seja entregue às queues apropriadas. O uso de bindings permite uma distribuição seletiva de mensagens com base em critérios específicos. Ao Exchange ExgResumo, do tipo direct, foi vinculada a queue Q_RESUMO, ficando assim com a estrutura demonstrada na figura 3.

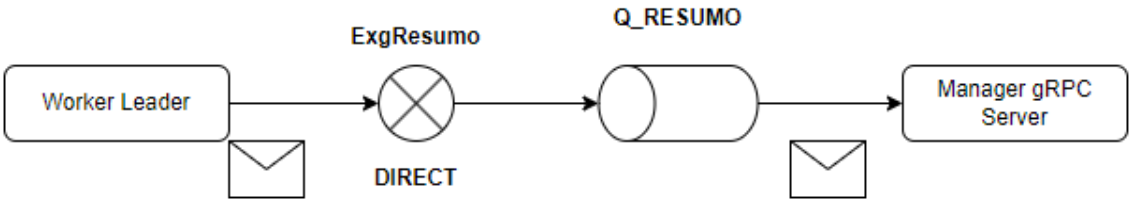

Figura 3 - Estrutura ExgResumo

Por sua vez, ao Exchange ExgSales, foram vinculadas as queues Q_ALIMENTAR e Q_CASA. Desta vez foram necessárias duas filas diferentes para distinguir as mensagens do tipo CASA e ALIMENTAR, através do Exchange do tipo TOPIC. Resultando no esquema apresentado na figura 4.

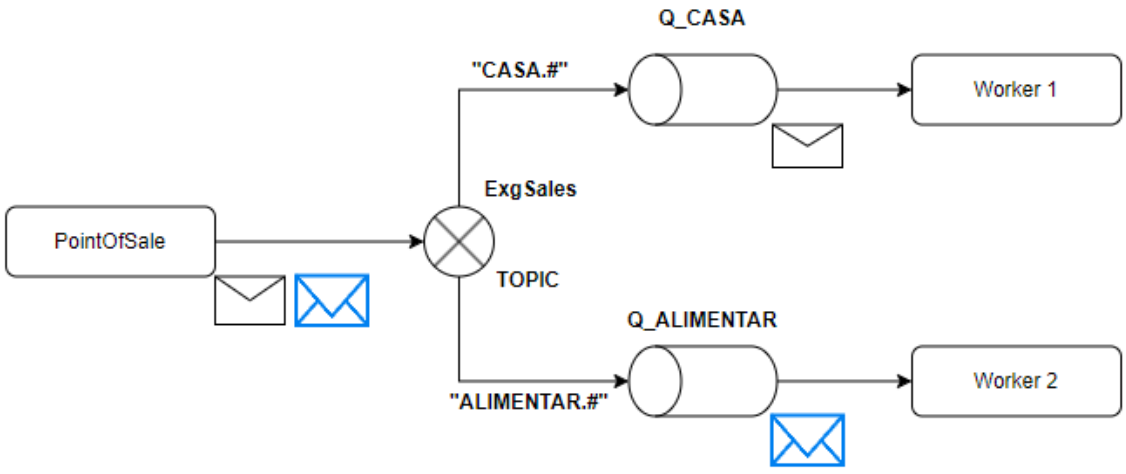

Figura 4 - Estrutura ExgSales

3. Gluster File System

O Gluster File System (GlusterFS) foi escolhido como o sistema de ficheiros distribuídos no sistema. Tem como principal função proporcionar a replicação eficiente de dados entre as máquinas virtuais (VMs), garantindo que os ficheiros, dos resumos dos Workers, estejam acessíveis e replicados em todas as Máquinas Virtuais.

Assim, os Workers, ao processarem as mensagens recebidas, escrevem essa informação nos respetivos ficheiros de resumo que são armazenados em diretórios partilhados no GlusterFS.

Para implementar esta funcionalidade, utilizamos os ips internos das vms e adicionamos umas às outras como peers. O que por sua vez permitiu criar um "glustervolume" e adicionamos as pastas de trabalho dos Workers ao volume, de forma a esta ser replicada em todas as VMs. Assim, quando um resumo é escrito, assim que o servidor gRCP de Manegment receber a mensagem com o nome do ficheiro, este pode efetuar o download do mesmo, com o conteúdo igual ao do ficheiro da VM que o criou.

```
IP do Broker: 34.175.54.4
      Menu
1 - Enviar 10 alimentos
2 - Enviar 10 produtos para casa
99 - Exit
Escolha uma opção:
```

Figura 5 - Interface de utilização do PointOfSales

4. Máquinas Virtuais na Google Cloud Platform

Múltiplas máquinas virtuais (VMs) foram alocadas na Google Cloud Platform (GCP) para executar diferentes componentes do sistema distribuído. Essas VMs fornecem a infraestrutura computacional necessária para hospedar aplicações como o PointOfSale, Workers, Broker, e outros, possibilitando a execução distribuída e escalável do sistema. As VMs colaboram para executar operações específicas, interagindo entre si conforme necessário.

São utilizadas as seguintes VMs:

- tf-node-1:

- tf-node-2:

- vm-lab-3:

Como configurações base, foram efetuadas em todas as VMs, as configurações recomendadas no enunciado, contando com o um Sistema Ubuntu v20.4, e2-medium e com as instalações do Java, Docker e Gluster.

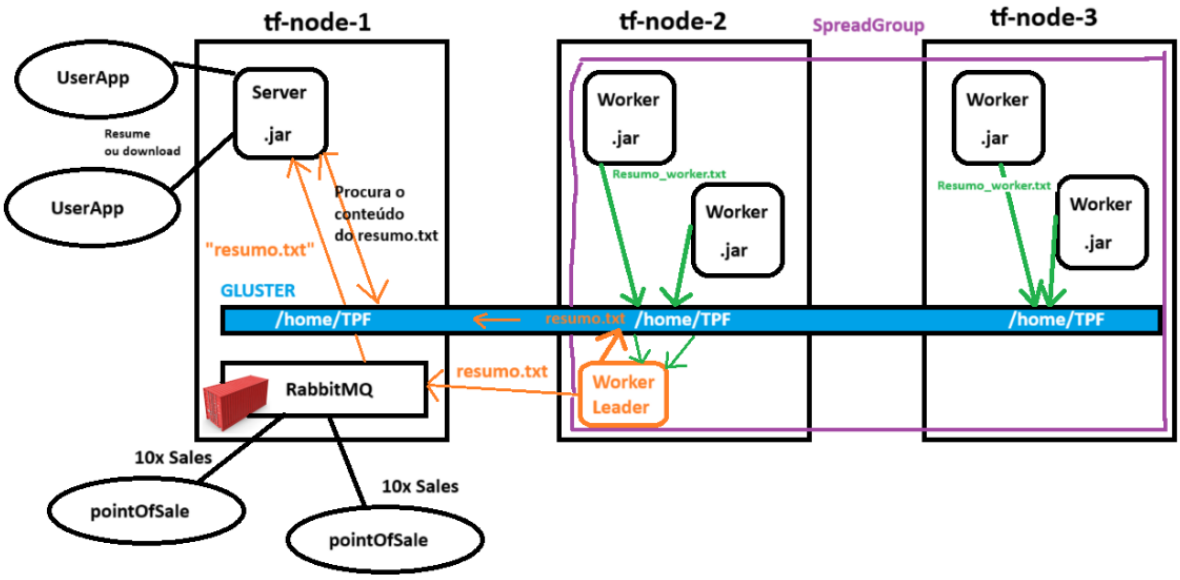

Figura 6 - Arquitetura das VMs

5. Comunicação em Grupo e Multicast (Spread)

O Spread é um framework utilizado para implementar a comunicação em grupo e multicast num sistema distribuído. Ele facilita a troca de mensagens entre os Workers, garantindo a coordenação eficiente durante operações críticas, como o resumo de vendas.

Os Workers pertencem a grupos Spread, o que permite a comunicação eficiente entre eles.

Assim, na nossa implementação optamos por criar dois grupos Spread. Um do servidor gRPC Manager (de nome “grpc”), e outro que engloba todos os Workers em funcionamento (de nome “SG”).

O grupo Spread “grpc” foi criado simplesmente para efetuar comunicações com o grupo “SG”, que assim nos permitiu enviar mensagens (“ALIMENTAR” ou “CASA”) do servidor para o grupo e estas mensagens serem recebidas simultaneamente por todos os Workers.

Já o grupo Spread “SG” para alem do objetivo de ser notificado pelo servidor e notificar todos os Workers, este grupo permite também os Workers de conhecerem a rede e os outros trabalhadores que estão a operar em conjunto com eles, o que é fundamental para por exemplo eleger um Líder para fazer o resumo das vendas, como será explicado mais à frente.

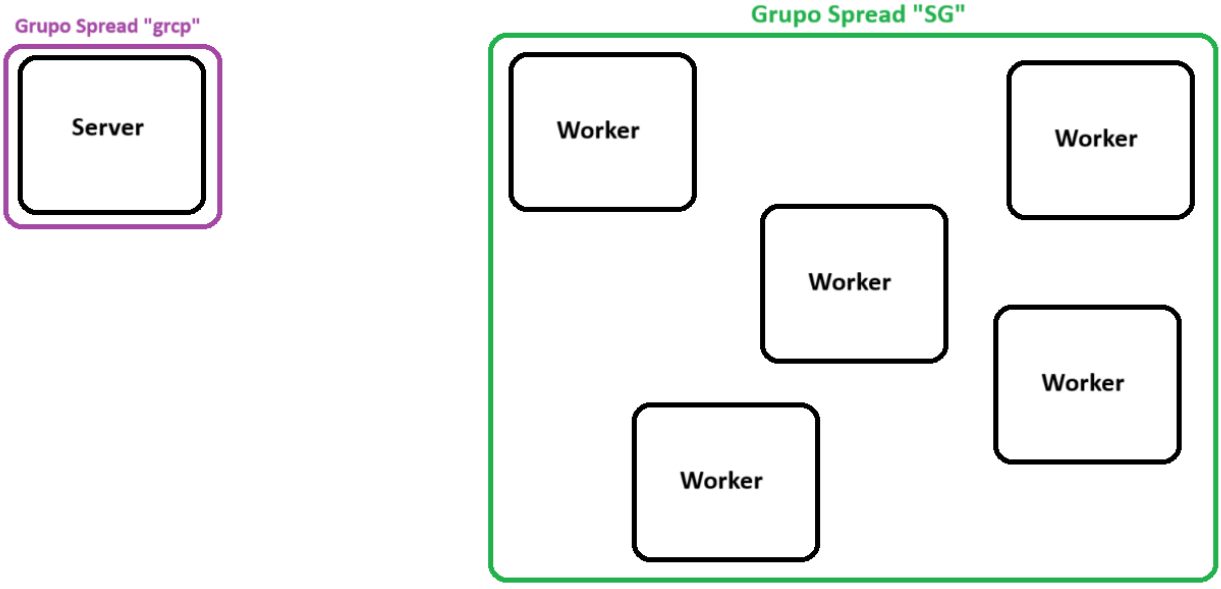

Figura 7 - Grupos Spread

b) Processamento de Mensagens e Escrita em Arquivos

Em relação à receção de mensagens, os Workers subscrevem as Queues do BrokerMQ, logo vão estar continuamente à espera que uma nova mensagem esteja na queue para ser consumida.

Assim, numa primeira fase, ao ligar um Worker, este deve ser configurado com os seguintes parâmetros:

- IP do Broker;
- IP do deamon do Spread;
- Um nome que o identifique no grupo Spread;
- O nome do grupo Spread a que se quer conectar;
- Qual a queue que vai querer subscrever;

Assim que configurado, o Worker fica apto para consumir mensagens do broker. Logo, assim que recebe uma mensagem, este deve escrever a mensagem num ficheiro de nome “resumo_nome_que_o_identifica.txt”, onde o nome que o identifica é também reconhecido por todos os membros do seu grupo Spread.

c) Ordem de Resumo de Vendas e Eleição do Líder

Em relação ao pedido de resumo de vendas e a eleição do líder, decidimos que a eleição é feita assim que é recebida uma mensagem no Grupo Spread “SG” do tipo “ALIMENTAR” ou “CASA”.

Ou seja, se o servidor notificar o grupo dos workers com a mensagem “ALIMENTAR”, os Workers que subscrevem a Queue “Q_ALIMENTAR” vão para de subscrever esta Queue e vão entre todos eleger um Lider.

O objetivo de eleger este Lider passa por escolher alguém para efetuar o resumo de todos os ficheiros que foram escritos até a esse momento. Assim, quando a mensagem vinda do Servidor gRCP chega ao grupo dos workers, estes verificam o conteúdo da mensagem e se for do seu interesse, executam a função performElection onde é passado o conjunto de membros do grupo que vao realizar a eleição.

Função **preformElection**:

- Em primeiro lugar param o consumo da queue ao mudar o valor da variável continueConsuming;
- Escolhem um leader ao chamar a função findLeader que vai ser explicada de seguida;
- Ao encontrarem o leader, se não forem escolhidas como leader continuam o consumo da Queue;
- Se forem escolhidas como leader, chamam a função writeElectionMessage que:
    - Escreve um ficheiro na sua diretoria a dizer que foi eleito;
    - Recorre à lista dos nomes indentificadores dos outros Workers do grupo para construir o nome dos ficheiros onde eles escreveram os seus resumos;
    - Para cada nome construído, lê o seu conteúdo e escreve em um novo ficheiro chamado “resumo.txt”.
- Executa a função restartConsumption que:
    - Retira o estatuto de Leader do Worker;
    - Apaga os resumos de todos os Workers da diretoria;
    - Volta ao consumo da Queue onde se encontrava conectado;

Função **findLeader:**

A função findLeader desempenha um papel importante pois é ela que define quem vai fazer o trabalho de resumir todos os resumos individuais dos Workers, no entanto esta função também acaba por ser simples, como foi sugerido, pois ela simplesmente:

- Recebe todos os membros do grupo que estão a fazer a eleição;
- Verifica o comprimento da lista;
- E escolhe o primeiro elemento como o Lider;
- Retorna o nome desse elemento;

No fim do processo de eleição e assim que os Workers retornam a consumir as Queues é também enviada uma mensagem de Publish para o broker, para a Queue “Q_RESUMO” com o nome do ficheiro onde esta este resumo. O que permite por sua vez o servidor gRPC subscrever o broker nesta Queue para obter o nome do ficheiro que deve entregar ao maneger.

Concluindo, a notificação do grupo “SG”, a escolha do Lider e a junção de todos os resumos, acabam por ser o ponto de junção de todas as componentes do trabalho, onde tantos as mensagens que vem dos pointOfSales como as notificações dos managers acabam por ser processadas e armazenadas.

d) Contrato do Servidor Manager gRPC

Por fim, mas não menos importante podemos também descrever a interação entre o manager e o servidor gRPC, que possibilita os managers de:

- Enviar um request resumeVendas, que envia uma string “ALIMENTAR” ou “CASA” dependendo da opção que o utlizador selecionar na interface;
- Receber a resposta do servidor ao request, que diz se a mensagem foi recebida ou se ocorreu algum erro;
- Enviar um request para fazer download do resumo;
- Receber o resumo numa diretoria especificada pelo mesmo;

Esta interação acaba por ser simples e permite o manager de resumir as vendas e fazer download de resumos.

Conclusão

O presente relatório explorou em detalhes a conceção, implementação e funcionamento de um sistema distribuído dedicado à simulação da operação de uma cadeia de supermercados. Ao longo dos capítulos, examinámos os componentes principais, desde a aplicação PointOfSale até a coordenação eficiente entre Workers, utilizando tecnologias como RabbitMQ, Gluster File System e gRPC.

O propósito deste projeto foi cumprido com sucesso, demonstrando a capacidade de integrar diferentes sistemas de interação e middleware em um ambiente distribuído. A implementação do sistema na Google Cloud Platform, a categorização e processamento de vendas, a comunicação em grupo, o multicast para resumos de vendas, a eleição do Leader e a coordenação via servidor Manager gRPC foram todos abordados de maneira abrangente. A utilização do Gluster File System garantiu a replicação eficiente de dados, crucial para a acessibilidade em todos os nós computacionais. A implementação do algoritmo de eleição foi destacada como um ponto crítico para o sucesso do sistema, e sua avaliação revelou eficiência na seleção do Leader, fundamental para a conclusão bem-sucedida dos resumos de vendas.

Em última análise, o sistema distribuído proposto não apenas atendeu aos requisitos funcionais, mas também ofereceu uma solução robusta, escalável e eficiente para os desafios enfrentados por cadeias de supermercados na gestão de grandes volumes de dados de vendas. Este projeto proporcionou uma oportunidade valiosa para a aplicação prática dos conceitos aprendidos durante a unidade curricular, destacando a importância dos sistemas distribuídos e da coordenação eficiente em ambientes complexos.