CD – Trabalho Prático de Avaliação Final

Uma imagem com texto, Tipo de letra, logótipo, Gráficos

Descrição gerada automaticamente

Relatório

Grupo08

Alunos: 48323 Simão Cabral

48268 Marçorio Fortes

52422 Marcelo Pereira

Índice

[**1.** **Introdução** 3](#_Toc179449313)

[**1.1** **Objetivo** 3](#_Toc179449314)

[**1.2** **Definições, Abreviaturas e Acrónimos** 3](#_Toc179449315)

[**1.3** **Referências** 3](#_Toc179449316)

[**1.4** **Organização do documento** 3](#_Toc179449317)

[**2.** **Posicionamento** 4](#_Toc179449318)

[**2.1** **Descrição do problema** 4](#_Toc179449319)

[**2.2 Posicionamento do produto** 4](#_Toc179449320)

[**3.** **Descrição de Participantes e do Ambiente de Utilização** 5](#_Toc179449321)

[**3.1 Resumo do utilizador** 5](#_Toc179449322)

[**3.3** **Resumo dos Parceiros** 5](#_Toc179449323)

[**3.4 Ambiente de Utilização** 5](#_Toc179449324)

[**4.** **Descrição Geral da Solução** 5](#_Toc179449325)

[**4.1 Resumo de Características** 5](#_Toc179449326)

[**4.1.1 Cliente** 6](#_Toc179449327)

[**4.1.2 Parceiro** 6](#_Toc179449328)

[**5.** **Restrições** 6](#_Toc179449329)

[**6.** **Níveis de qualidade** 6](#_Toc179449330)

[**7.** **Precedências e Prioridades** 6](#_Toc179449331)

[**8.** **Outros requisitos aplicáveis** 6](#_Toc179449332)

[**8.1 Requisitos do sistema** 6](#_Toc179449333)

[**9.** **Requisitos de Documentação** 7](#_Toc179449334)

[**9.1 Manual de Utilização** 7](#_Toc179449335)

[**9.2 Ajuda Online** 7](#_Toc179449336)

Relatório

1. **Introdução**

Neste documento descreve-se o sistema implementado, permitindo de forma que se possa compreender o objetivo, pressupostos, a arquitetura, a configuração para execução do sistema distribuído, que tira uso dos *middleware* estudados e já utilizados nos Laboratórios das aulas práticas.

* 1. **Definições, Abreviaturas e Acrónimos**

Middleware - Software intermédio que conecta e permite a interação entre diferentes sistemas ou aplicações.

* 1. **Referências**

Documentação de apoio à unidade curricular Computação Distribuída

* 1. **Organização do documento**

O documento está organizado da seguinte forma:

* Secção 1: secção de introdução;
* Secção 2: secção onde é apresentado os contratos entre os clientes e os servidores presentes no grupo;
* Secção 3: secção onde é apresentado o funcionamento do *Register* e servidores dentro do grupo;
* Secção 4: secção onde é apresentado funcionamento das aplicações com o espaço de ficheiros partilhados;
* Secção 5 a 9: secções onde é descrito o funcionamento entre as aplicações (*Workers*) com o *RabbitMQ*.

1. **Cliente-Servidor**

****

**Figura 1**

## **2.1** **Contrato GRPC**

Para concretização da solução, foram implementados 2 contratos:

* Contrato do cliente com as instâncias SVC (defenir melhor o nome);
* Contrato do cliente com o servidor Register

### **2.1.1 Cliente-Register**

O contrato tem um único método unário utilizado pelo cliente na obtenção de IP e porto de um servidor SVC figura 2. O corpo da mensagem pode ser verificado na figura 3.

Uma imagem com texto, Tipo de letra, captura de ecrã

Descrição gerada automaticamente

Figura 2

Uma imagem com texto, Tipo de letra, captura de ecrã, número

Descrição gerada automaticamente

Figura 3

### **2.1.2 Cliente-SVC**

O contrato tem dois métodos:

* uploadImage – Com stream de cliente para o upload das imagens em blocos de bytes;
* downloadImage – Com stream de servidor para download das imagens em blocos de bytes;

O mais relevante no contrato é a estrutura das mensagens. Para a mensagem ImageBlock tira-se proveito do tipo Oneof, para o envio dinâmico dos metadados da imagem (ImageData), e os bytes que constituem a mesma (block), como verificado na figura 4.

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Figura 4

Desta forma o servidor pode iniciar a escrita da imagem no espaço partilhado, sem o processo ocupar a memória com todo conteúdo da imagem.

1. ***Register* e servidores dentro do grupo**

## **3.1 *Register***

O ***Register*** é o primeiro membro do grupo e tem como função apenas receber mensagens enviadas pelos servidores (***svc***). Utilizamos a implementação do ***AdvancedMassageListener*** que permite lidar com as mensagens enviadas pelos elementos do grupo e mensagens do spread de forma independente. Assim que um serviço novo entra no grupo, é enviada a mensagem automática de ***menbership*** e uma mensagem contruída por nós contendo o nome, ip e porto do serviço que se juntou ao grupo. Assim que esse serviço entra no grupo é guardado numa estrutura de dados chamada ***SvcRegisterManager*** que contem uma lista de serviços e tem os métodos para gerir os mesmos. Da mesma forma, quando um serviço sai do grupo, o mesmo será removido da lista de serviços disponíveis.

Uma forma que encontramos para controlar os serviços com mais pedidos foi ter, na estrutura de dados responsável por representar o serviço (***ServiceInfo***), uma variável que conta quantas vezes o serviço enviou, na mensagem de grupo, uma mensagem com o parâmetro *estado* (***imageStatus***) não nulo e com o estado ***Downloading*** ou ***Uploading***. Deste modo o ***Resgister Service*** consegue escolher o serviço que tem menos pedidos de ***Downloading*** ou ***Uploading*** feitos no grupo. Escolhemos esta solução para facilitar e porque é mais logico do que apenas somar o número de clientes que associamos a cada serviço, pois estes podem nem realizar pedidos e desligarem a ligação, ou fazerem vários pedidos. Deste modo achamos que existe um melhor balanceamento. A melhor solução seria controlar quantos pedidos de ***download*** e ***uploanding*** os serviços estavam a realizar no momento que o cliente pedisse um serviço e escolhêssemos o com menos pedidos. Mas pela dificuldade da solução optamos pela que já foi referida.

**3.2 Servidores dentro do grupo**

Os servidores de grupo elegem o líder de

1. **Aplicações com o espaço de ficheiros partilhados**

## **4.1**