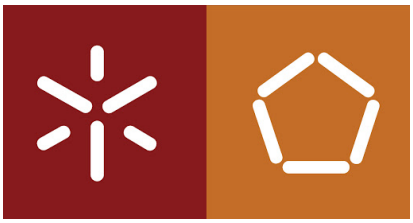

Universidade do Minho

Licenciatura em Engenharia Informática

Comunicação por Computadores

TP2: FolderFastSync - Sincronização rápida de pastas em ambientes *serverless* Grupo 62

Rodrigo Rodrigues (A93201) David Duarte (A93253)
João Machado (A89510)

Ano Letivo 2021/2022

1 Introdução

Este documento é um relatório técnico sobre o trabalho desenvolvido no âmbito do TP2 da Unidade Curricular *Comunicação por Computadores*.

Neste relatório apresentamos o protocolo desenvolvido perante a proposta de desenvolver uma aplicação de sincronização rápida de pastas sem necessitar de servidores nem de conectividade internet, designada por *FolderFastSync* **(FFSync)**.

O projeto, desenvolvido na linguagem de programação Java, consistiu na criação de um protocolo de dados. Através deste protocolo, dois ou vários clientes são capazes de sincronizar duas pastas entre dispositivos. Estas pastas estão localizadas nos seus respetivos dispositivos, que comunicam com utilizando esse mesmo protocolo desenvolvido pelo grupo de trabalho que funciona sobre UDP. Para além dos módulos constituintes do Java por defeito, recorreu-se ao uso de vários modulos packages, sockets entre outros que iriam permitir receber e responder a pedidos HTTP de forma assíncrona e paralela.

2 Especificação do Protocolo

2.1 Formato das Mensagens Protocolares (PDU)

Foi definido um formato base de todas as mensagens trocadas no sistema (PDU - Protocol Data Unit).

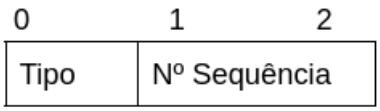

Figura 1: Formato base de uma mensagem

O formato base de uma mensagem contém apenas:

- Tipo: tipo de mensagem
- Nº Sequencia

Existem os seguintes **tipos** de mensagens:

- **1** : Ack
- **2** : Transferência de Dados
- **3** : Fim de Conexão
- **4** : Metadados
- **5** : Pedido de Ficheiro

2.2 Função e Significado dos campos

Mensagens Ack As mensagens **Ack** têm o seguinte formato:

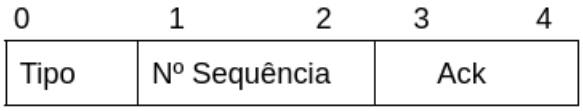

Figura 2: Formato de uma mensagem **Ack**

- Tipo: 1 (Consts.ACK)
- Nº Sequencia
- Ack: Nº sequencia do pacote que pretende confirmar

Mensagens de Dados As mensagens **Dados** têm o seguinte formato:

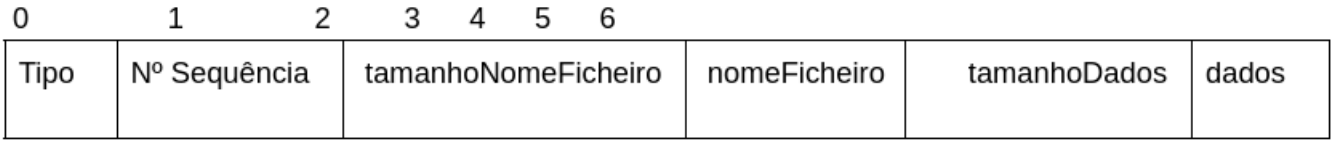

Figura 3: Formato de uma mensagem **Dados**

- Tipo: 2 (Consts.TRANSFERENCIA_DADOS)
- Nº Sequencia
- Tamanho do Nome do Ficheiro
- Nome do Ficheiro
- Tamanho dos Dados a transferir
- Dados a transferir

Mensagens de Metadados As mensagens **Metadados** têm o seguinte formato:

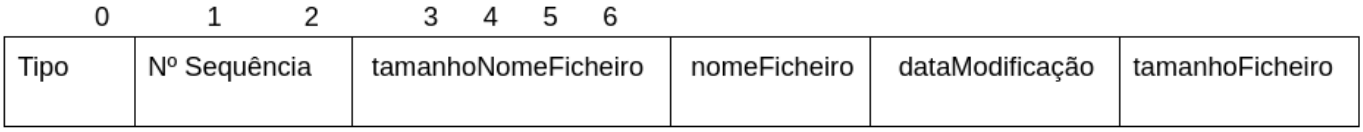

Figura 4: Formato de uma mensagem **Metadados**

- Tipo: 4 (Consts.META_DADOS)
- Nº Sequencia
- Tamanho do Nome do Ficheiro
- Nome do Ficheiro
- Data de modificação
- Tamanho do ficheiro

Mensagens de Pedido de Ficheiro As mensagens **Pedido de Ficheiro** têm o seguinte formato:

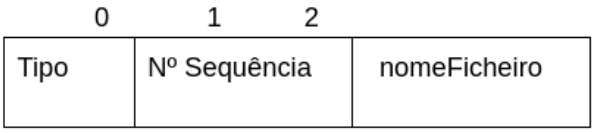

Figura 5: Formato de uma mensagem **Pedido de Ficheiro**

- Tipo: 5 (Consts.PEDIDO_FICHEIRO)
- Nº Sequencia
- Nome do Ficheiro

2.3 Interações

Existem 3 fases principais no serviço: Estabelecimento de conexão, transferência de ficheiros e fim de conexão. Na primeira fase, o cliente envia um pedido ao servidor para iniciar conexão e deve obter uma resposta afirmativa (Three-way handshake).

Assim que está estabelecida a conexão, inicia-se a 2ª fase : transferência de ficheiros. Primeiro, o cliente faz um pedido de listagem de ficheiros e, de seguida, faz um pedido de Metadados (para averiguar se deve requisitar a transferência de determinados ficheiros, i.e, por exemplo, no caso de haver ficheiros com o mesmo nome, mas com datas de edição diferentes, então deve pedir o ficheiro). Uma vez que conhece os ficheiros disponíveis para download e os seus metadados, o cliente pede os ficheiros que são diferentes àqueles que possui e o servidor envia-os. Quando o cliente estiver na posse dos ficheiros pretendidos, o processo repete-se, mas para o servidor.

Por fim, a 3ª fase corresponde ao término de conexão, em que o cliente pede para terminar a conexão e o Servidor responde com Ack.

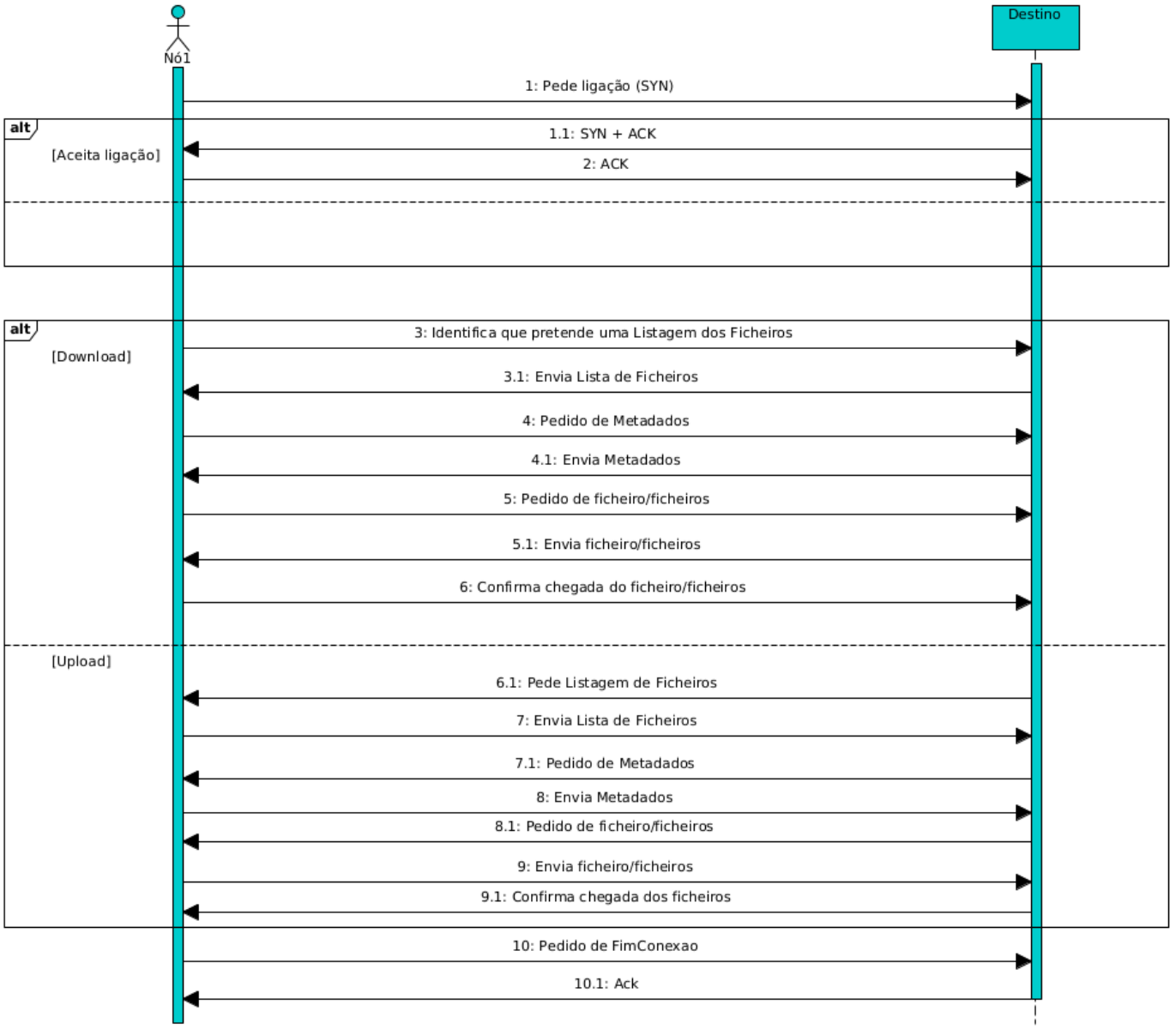

Figura 6: Diagrama de Sequência

3 Implementação

Este protocolo e as repetivas aplicações foram implementados a partir da linguagem de programação **JAVA**.

3.1 Package Pacotes

Na package **Pacotes** encontram-se os diversos datagramas previstos para o protocolo bem como as classes que permitem a formatação correta dos datagramas para que possam ser enviados e recebidos com o formato pretendido.

- PedidoConexao
- Ack
- Metadados
- PedidoFicheiro
- Dados (transferência de dados)
- FimConexao

Nestas classes encontram-se os construtores e os métodos ***serialize*** que constroem os *DatagramPackets* a serem enviados, bem como todos os outros métodos necessários.

3.2 Package App

Na package **App** encontram-se as classes:

- FFSync (classe principal)
- Client
- Server
- ClientHandler

A classes Client e Server correspondem às classes que lidam com o lado do cliente e do servidor, respetivamente. A classe ClientHandler é a classe que processa os pacotes que o servidor recebe.

4 Resultados

5 Conclusão

Com a realização deste trabalho fomos capazes de implementar de forma prática vários conceitos lecionados nas aulas de Comunicações por Computador, o que nos permitiu reconhecer a sua utilidade importância no mundo real. Alguns aspetos foram mais complicados e infelizmente não fomos capazes de cumprir os requisitos obrigatório, de forma geral estamos completamente insatisfeitos com o que fomos capazes de conceber que sofreu bastante com o facto de não termos tido Redes e Sistemas Distribuídos que dão luzes essenciais para as fundações deste trabalho.