

DropMusic

Arquivo de música com partilha de ficheiro

Relatório meta 1

Sistemas Distribuídos | 27 de outubro 2018

Indice

[**Introdução** 2](#_Toc528438224)

[**Arquitetura de Software** 2](#_Toc528438225)

[**Consola de administração e cliente rmi** 3](#_Toc528438226)

[**Servidor rmi** 4](#_Toc528438227)

[**Servidor multicast** 4](#_Toc528438228)

[**Distribuição de tarefas** 4](#_Toc528438229)

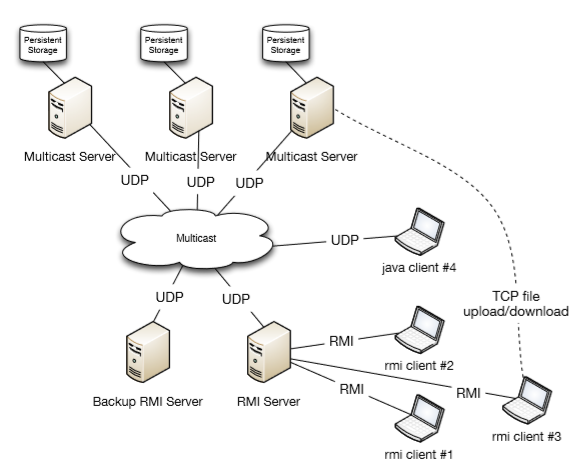
[**Testes de software** 5](#_Toc528438230)

## **Introdução**

Este projeto tem como objetivo criar um sistema de gestão de partilha de músicas. O sistema tem todas as informações relevantes sobre as músicas, tais como autores, álbuns, grupos musicais, compositores ou outros que se considere importantes. O sistema tem utilizadores que acedem às funcionalidades com a aplicação DropMusic. Alguns utilizadores são também editores que podes acrescentar e modificar informações.

## **Arquitetura de Software**

A aplicação DropMusic consiste em três programas: servidor RMI, servidor Multicast e cliente RMI. Os servidores RMI são idênticos embora seja inicialmente primário e outro secundário. Os servidores Multicast, são idênticos exceto possíveis questões de configuração. O Cliente RMI liga-se aos servidores RMI e permite aos utilizadores acederem ao DropMusic.



## **Consola de administração e cliente rmi**

Inicialmente, a consola, ou seja, o cliente RMI, tenta ligar-se ao servidor RMI através do lookup. Seguidamente, através do comando registar ou login tenta-se aceder à aplicação, sabendo que só tendo uma conta na DropMusic poderá aceder às informações que esta contém.

* Para o login: “login ; username ; palavra\_passe”
* Para o registo: “registar ; username ; palavra\_passe”

Caso o utilizador seja regular tem as funcionalidades:

1. Pesquisar música;
2. Consultar detalhes álbum;
3. Consultar detalhes artista;
4. Upload de música;
5. Download de música;
6. Partilhar uma música;
7. Logout.

Caso o utilizador seja editor, tem as seguintes funcionalidades:

1. Pesquisa música;
2. Gerir artistas;
3. Gerir álbuns;
4. Gerir músicas;
5. Dar privilégios de editor a um utilizador;
6. Consultar detalhes álbum;
7. Consultar detalhes artista;
8. Upload de música;
9. Download de música;
10. Partilhar uma música;
11. Logout.

A gestão de músicas, álbuns e artistas inclui adição, remoção e alteração dos mesmos.

## **Servidor rmi**

O servidor RMI implementa a interface RMI que especifica os métodos das funcionalidades implementadas.

Com o mesmo código teremos um servidor primário e um servidor secundário em caso de falha do servidor primário. Assim, para o caso do servidor primário se voltar a conectar passa a ser o secundário.

Começa por verificar se já existe um servidor RMI no rmiregistry (com porto 7000), caso isso aconteça fica como backup, caso contrário fica como principal.

O servidor secundário envia continuamente mensagens para verificar se o primário está “alive”.

O servidor RMI aguarda ligações por parte da consola de utilizador normal ou editor que a ele tentam fazer login, registo ou praticar alguma funcionalidade consoante os privilégios.

## **Servidor multicast**

Começa por carregar os ficheiros de objeto, que será a nossa base de dados, ou então cria-os caso não tenham sido criados.

Todos os dados da aplicação são acedidos unicamente pelo servidor multicast.

Para enviar uma resposta ou receber uma resposta é feito por uma SocketMulticast, que tem como multicast address 224.3.2.1 e porto 5000;

Para distinguir as diferentes funções atribuímos números com atributos separados por ‘;’.

## **Distribuição de tarefas**

O trabalho foi igualmente distribuídos pelos 2 elementos do grupo. Inicialmente um elemento focou-se no ServidorRMI e ClienteRMI enquanto que o outro se focou no ServidorMulticast. Após estas estarem implementadas e ter feito a junção trabalhamos o projeto foi desenvolvido como um todo à vez. No entanto, independentemente das divisões de tarefas, trabalhamos em conjunto, ajudando-nos mutuamente.

## **Testes de software**

|  |  |
| --- | --- |
| Requisitos Funcionais |  |
| Registar novo utilizador | PASS |
| Login protegido com password | PASS |
| Introduzir artistas, álbuns e músicas | PASS |
| Pesquisar álbuns por artista e por título de álbum | FAIL |
| Consultar detalhes de álbum (incluindo músicas e críticas) | PASS |
| Editar detalhes de álbum (incluindo músicas) | FAIL |
| Escrever crítica sobre um álbum (com pontuação) | PASS |
| Consultar detalhes de artista (e.g., discografia, biografia) | PASS |
| Dar privilégios de editor a um utilizador | PASS |
| Notificação imediata de privilégios de editor (online users) | FAIL |
| Notificação imediata de re-edição de descrição de álbum (online users) | FAIL |
| Entrega posterior de notificações (offline users) | FAIL |
| Upload de ficheiro para associar a uma música existente | FAIL |
| Partilhar um ficheiro musical e permitir o respetivo download | FAIL |
| Tratamento de Exceções |  |
| Avaria de um servidor RMI não tem qualquer efeito nos clientes | PASS |
| Não se perde/duplica músicas se os servidores RMI falharem | PASS |
| O serviço funciona desde que haja um servidor multicast disponível | PASS |
| Avarias temporárias (<30s) dos N servidores são invisíveis para clientes | PASS |
| Pedidos são sempre processados por N>=1 servidores multicast | FAIL |
| Pedidos de leitura são respondidos apenas por um servidor multicast | PASS |
| Failover |  |
| Servidor RMI primário testa periodicamente o secundário | PASS |
| Em caso de avaria longa os clientes RMI ligam ao secundário | PASS |
| Servidor RMI secundário substitui o primário em caso de avaria longa | PASS |
| O failover é invisível para utilizadores (não perdem a sessão) | PASS |
| O servidor original, quando recupera, torna-se secundário | PASS |
| Relatório |  |
| Arquitetura de software detalhadamente descrita | PASS |
| Detalhes do funcionamento do servidor UDP (protocolo multicast, etc.) | FAIL |
| Detalhes do funcionamento do servidor RMI (interface, javadocs, etc.) | PASS |
| Distribuição de tarefas pelos elementos do grupo | PASS |
| Testes de software (tabela com descrição + pass/fail de cada teste) | PASS |
| Extra (até 5 pontos) |  |
| Servidores multicast replicados mantêm os dados sincronizados (5p) | FAIL |
| Balanceamento da carga dos servidores (3p) | FAIL |
| STONITH nos servidores RMI (3p) | FAIL |
| Outros (a propor pelos alunos) | FAIL |
| Pontos Obrigatórios |  |
| Pontualidade (-10) |  |
| O projeto corre distribuído por várias máquinas (-5) |  |
| Configuração não requer recompilação (-5) |  |
| A aplicação não apresenta erros/exceções/avarias (-5) |  |
| Código legível e bem comentado (-5) |  |
| No dia da defesa |  |
| Os estudantes chegam 15 minutos antes para prepararem tudo |  |
| Trazem duas máquinas (e.g., dois portáteis) |  |
| Portátil #1 corre 1 servidor multicast e 1 cliente RMI |  |
| Portátil #2 corre 1 servidor multicast, 2 servidores RMI e 1 cliente RMI |  |
| Há dados de teste (3 utilizadores, 3 álbuns e várias músicas) |  |