FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Relatório DropMusic

Sistemas Distribuídos

2018/2019

Cláudio André Ventura Alves 2016225581

Diogo Alexandre Cardoso Semedo 2016225626

Meta 1

Neste projeto criámos um multicast server que guarda a informação na sua base de dados, e que comunica com um rmi server através de pacotes udp. O nosso rmi server apresenta também um rmi backup, este comunica através de rmi, para se manterem atualizados. O cliente rmi comunica com o rmi server através de rmi. Caso seja feita uma operação de upload ou download é feita uma ligação de tcp direta entre o cliente e o multicast server.

O servidor multicast e o rmi server comunicam com pacotes que contêm um hashmap. Estes hashmaps tem várias chaves e valores que permitem o correto funcionamento do programa, através de um protocolo definido. Este encontra-se comentado nos vários métodos. O servidor multicast tem também uma thread que comunica com o servidor rmi para saber que multicasts servers estão ligados, de maneira a serem identificados.

O rmi server principal testa o rmi server backup periodicamente, quando o rmi server principal vai abaixo o rmi server backup assume o papel de rmi server principal. Quando o backup assume o papel de rmi principal os clientes mantêm a sessão aberta, caso esta esteja aberta. O rmi principal quando altera as suas variáveis temporais actualiza no rmi server backup caso este esteja online.

A distribuição de tarefas nesta meta no projeto foi a sugerida pelo docente da cadeira, um elemento focou-se mais na parte de multicast e o outro elemento focou-se mais no rmi, mas ambos em caso de dúvidas de implementação conversaram entre si.

	A	B	C
1	Testes	PC 1 (1MS,2RMIS,1RMIC)	PC 2 (1MS, 1RMIC)
2	Resgisto	pass	pass
3	Login	pass	pass
4	promote user online	pass (recebeu notification)	pass (deu promote)
5	promote user offline	pass(deu promote)	pass(recebeu notification quando deu login
6	insert artist	pass	pass
7	inset album	pass	pass
8	insert music	pass	pass
9	show all artist	pass	pass
10	show all album	pass	pass
11	show all music	pass	pass
12	show details artist	pass	pass
13	show details album	pass	pass
14	show details music	pass	pass
15	write review	pass	pass
16	search music genre	pass	pass
17	search music artist	pass	pass
18	search music album	pass	pass
19	edit --> notifications on e off	pass	pass
20	remove artist	pass	pass
21	remove album	pass	pass
22	remove music	pass	pass
23	edit values	pass	pass
24	upload	pass	pass
25	share	pass	pass
26	download	pass	pass

Meta 2

Arquitetura

O projeto web é composto por models, actions, websocket e jsp, tudo integrado com struts2, ou seja, seguindo um modelo MVC.

Nos models temos a classe User onde fazemos a conexão com o RMIServer, onde estão todos os métodos para a comunicação com o RMIServer, também temos a classe DropBoxRestClient onde estão os métodos que criam e tratam os http request para a DropBox Api.

Nas actions temos três actions diferentes, a de login, registo e a MainAction esta contém todas as actions relevantes para o bom funcionamento dos menus.

Na websocket temos uma conexão com o RMIServer, este controla tudo o que é notificações.

Nas jsp com a ajuda das tags de struts2 geramos todos os menus em html.

Resumindo, apresentamos uma aplicação web com um servidor https, que atua como um cliente RMI para com o servidor RMI. A comunicação instantânea é feita a partir de websockets permitindo ao cliente receber notificações e alteração de valores em tempo real.

(Nota: Criámos um certificado ssl, para correr o webserver em https.)

Integração de Struts2 com o Servidor RMI

Esta integração é feita através do model User que integra a conexão com o rmiserver o que lhe permite utilizar os métodos de comunicação com a base de dados através do rmiserver. Quando o utilizador faz login é criado uma instância da classe User que é guardada na session do browser, depois qualquer action que necessite de métodos do RMIServer chama-os através da instância da classe User.

Integração de websockets com struts2 e RMI

Esta integração é feita em cada página que é carregada, abre uma conexão de websocket que pode ser usado caso haja alguma notificação para enviar ao utilizador. A conexão é feita no JavaScript da jsp.

Integração de REST Webservices (Dropbox)

Como pedido pelo docente foi feita uma integração com a API da Dropbox, a comunicação com a API é feita através da classe DropBoxRestClient. Foi feita uma autenticação OAuth com recurso à biblioteca Scribejava.

50	Requisitos Funcionais	50	
5	Registar novo utilizador**	5	Pass
5	Login protegido com password (acesso a todas as páginas)	5	Pass
5	Introduzir artistas, álbuns e músicas	5	Pass
5	Remover artistas que não tenham álbuns nem músicas	5	Pass
5	Pesquisar álbuns por artista e por título de álbum	5	Pass
5	Consultar detalhes de álbum (incluindo músicas e críticas)**	5	Pass
5	Editar detalhes de álbum (incluindo músicas)	5	Pass
5	Escrever crítica sobre um álbum (com pontuação)	5	Pass
5	Consultar detalhes de artista (e.g., discografia, biografia)	5	Pass
5	Dar privilégios de editor a um utilizador**	5	Pass
15	WebSockets	15	
5	Notificação imediata de privilégios de editor**	5	Pass
5	Notificação imediata de re-edição de descrição textual**	5	Pass
5	Atualização imediata da pontuação média das críticas	5	Pass
25	REST	25	
5	Associar conta de utilizador à Dropbox	5	Pass
5	Associar ficheiro contido na Dropbox a uma música existente	5	Pass
5	Partilhar música através da Dropbox	5	Pass
5	Tocar uma música diretamente na página do DropMusic	5	Pass
5	Login com a conta da Dropbox	5	Not Implemented
	Extra (até 5 pontos)	4	
4	Utilização de HTTPS (4 pts)	4	Pass