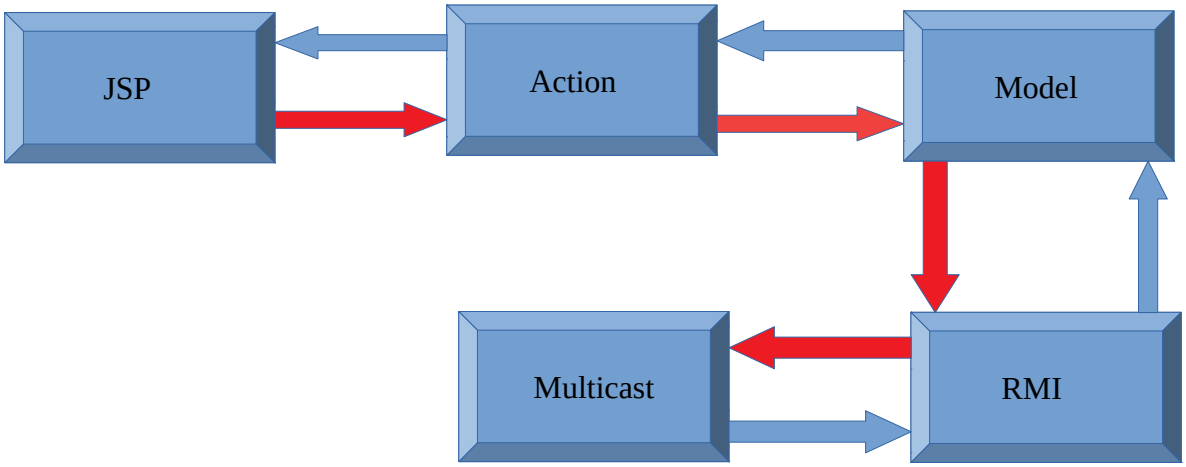
Projeto final de Sistemas Distribuídos

UNIVERSIDADE DE COIMBRA

António Lopes - 2017262466
Lucas Ferreira - 2016243439

Arquitetura Software



Nesta meta desenvolvemos uma aplicação que usa o Tomcat como servidor(HTTPS) que comunica com servidor RMI e o servidor Multicast desenvolvidos na primeira meta, e dispõe os dados recebidos deste numa página web.

O servidor é constituido por Models, Views/Actions , Controllers e Websockets. Cada Action vai chamar os métodos necessários do Model, que comunica com o RMI e devolve os dados obtidos para as actions, e após o tratamento destas verifica se são validas ou não, a informação será depois apresentada numa pagina web . As rotas do tratamento das Actions encontram-se definidas num ficheiro struts. Os websockets serão usados para notificar users, em tempo real , se recebem direitos de admin ou não, e para atualizar a pagina de administrador.

O servidor RMI recebe pedidos por parte do Model e envia para o Servidor Multicast a informação desses pedidos. Cabe também ao servidor RMI reenviar a informação obtida pelo Multicast para o Model.

Integração do Struts2 com o Servidor RMI

Nesta meta, a integração do Struts2 com o RMI passa por processar os pedidos como uma “action”, ou seja, a qualquer pedido de um cliente nos ficheiros jsp é chamada uma ação correspondente, que por sua vez interage com o RMI. Para tal, temos um bean chamado “HeyBean” que nos permite aceder diretamente aos métodos equivalentes a um cliente RMI para comunicar com o servidor. Ao usar o ficheiro “struts.xml” é feita ligação entre os ficheiros jsp e as actions, que por sua vez processam os pedidos do cliente.

Heybean

- getUsername: Getter do username do cliente.

- traduz: Método que recebe uma arraylist com a informação sobre o título, url e texto de uma página, devidamente traduzido, acedido remotamente pelo método translate.

- logout: Método que recebe o username do cliente e chama remotamente o método de logout no servidor RMI.

- loginFB: Método que recebe um token associado à API do facebook que por sua vez é remotamente acedido pelo método LoginFacebook.

- getLoginAuth: Método que recebe o link do facebook para conseguir obter os dados de um dado utilizador do facebook.

- mysearch: Método que recebe o username de um cliente e lhe devolve o histórico desse mesmo cliente.

- checkAdmins: Método que invoca a lista de utilizadores e administradores correspondentes no servidor.

- giveAdmin: Método que permite, ou não, dar permissões a um utilizador ser administrador.

- info: Método que ordena a lista de sites pelo número de referências.

- getIndexedPages: Método que procura um dado site e os links referentes a ao mesmo.

- indexNewUrl: Método que recebe o username e um URL, de modo a indexá-lo e quem o indexou.

- login: Método que permite ao utilizador fazer login.

- search: Método que permite a um utilizador fazer uma pesquisa referente à palavra pedida.

- register: Método que permite a um utilizador anónimo proceder a um registo.

- notification: Método que verifica se um utilizador tem notificações offline, ou não, de permissões de administrador.

- setNotification: Método que associa uma notificação a um dado utilizador.

- checkState: Método que permite verificar o estado de ligação do utilizador no servidor.

- connectFb: Método que permite a um utilizador conetar-se ao facebook.

- getConnectAuth: Método que recebe o link de autorização para um user existente ao facebook.

Actions/JSP

- adminPageAction: Ação que invoca o método execute, que por sua vez é chamado pelo adminPage.jsp que mostra a página de um administrador e as opções relativas a este.

- GiveAdminAction: Ação que invoca o método execute, que por sua vez é chamado pelo giveadmins.jsp que mostra a página onde um administrador pode dar permissões de administrador a um utilizador.

- HistoricoAction: Ação que invoca o método execute, que por sua vez é chamado pelo history.jsp que mostra a um utilizador o histórico de pesquisas deste.

- indexedPages: Ação que invoca o método execute, que por sua vez é chamado pelo searchIndexes.jsp que mostra a um utilizador as páginas mais indexadas.

- indexUrlAction: Ação que invoca o método execute, que por sua vez é chamado pelo index.jsp onde permite a um utilizador fazer a indexação de um site.

- LoginAction: Ação que invoca o método execute, que por sua vez é chamado pelo login.jsp que permite ao utilizar fazer login.

- logoutAction: Ação que invoca o método execute, e que remove os dados de sessão de um dado utilizador.

- PrintAdminAction: Ação que invoca o metodo execute, que mostra a um utilizador o número de utilizadores.

- RegisterAction: Ação que invoca o método execute, que por sua vez é chamado pelo register.jsp que permite ao utilizador registar-se na plataforma.

- SearchUrl: Ação que invoca o método execute, que por sua vez é chamado na index.jsp onde permite ao utilizador fazer uma pesquisa.

- switchPageAction: Ação que invoca o método execute, que leva o utilizador à página de login.

- translateAction: Ação que invoca o método execute, onde permite ao utilizador traduzir títulos e textos de páginas.

Websockets

Os websockets implementados nesta meta possuem duas listas onde contêm as sessões e usernames de cada utilizador, adicionados e removidos à mesma medida. No método correspondente ao OnOpen, este adiciona a sessão e verifica quais delas estão inativas posteriormente, removendo as sessões e os users correspondentes. Já no OnMessage, esta faz set do username correspondente e adiciona o utilizador à lista, dado que o websocket está constantemente a receber o nome dos users que estão online. Quando há a necessidade de enviar alguma notificação, é chamado o método sendMessage que por sua vez envia em texto para o utilizador correto.

REST

Neste trabalho usamos dois tipos de REST APIS, o Facebook e o Yandex.

Relativamente ao Facebook usamos autenticação Oauth, ao iniciar um serviço onde inserimos a key da aplicação gerada de login de facebook, assim como a secret key deste , o link que será usada para redirecionar a pagina para uma action, após o user verificar autirozacao no facebook,(callback) e as permissoes que desejamos obter no scope. Após preenchermos os parâmetros necessários obtemos o url, que vai enviar o user para uma pagina de aceitaçao de partilhas dos dados do facebook. Quando o user dá acesso da partilha dos seus dados,é enviado para uma action que envia um token gerado pelo facebook para o RMI. Este token será usado para aceder ao id e ao nome do user, que serao usados para inserir novo utilizador na base de dados, ou no caso do user já se encontrar na bases de dados confirma o login, permitindo assim o login via Facebook.

Seguindo o mesmo raciocínio , criamos também uma função que permite a um utilizador registado na aplicação associar o seu facebook a uma conta do UcBusca.

No caso do Yandex API , criamos uma classe nomeada de Yandex que possui as funções que irão ser utilizadas para a tradução das pagina. Esta classe e composta por: translate, que recebe o texto que ira ser traduzido, a língua original e a língua que pretendemos traduzir, o detect para detetar a língua original do texto.

Relativamente a sua integração com RMI criámos 2 tipos de funcões , ambas recebem uma arraylist com a informação dos sites . A primeira destas funções nomeada de translate irá traduzir o titulo e texto das paginas. A class Yandex é chamada para detetar a linguagem do texto em questão e consequentemente traduzir a o texto para Português, inserindo o resultado numa Arraylist que irá ser retornada . A outra função nomeada de detetaLínguas serve para devolver uma Arraylist com as informaçoes normais de um site mas acrescenta a língua em se encontra escrita para mais tarde ser exibida no ecrã.

Testes feitos à plataforma

Sistemas Distribuídos 2019/20 - Meta 2	
Nome:	
Número de Aluno:	
Nota Final:	
Registar novo utilizador**	Pass
Acesso protegido com password (todas as páginas exceto pesquisas)	Pass
Indexar novo URL introduzido por administrador	Pass
Indexar iterativamente ou recursivamente todos os URLs encontrados	Pass
Pesquisar páginas que contenham um conjunto de palavras	Pass
Resultados ordenados por número de ligações para cada página	Pass
Consultar lista de páginas com ligações para uma página específica	Pass
Consultar lista de pesquisas feitas pelo próprio utilizador**	Pass
Dar privilégios de administrador a um utilizador**	Pass
Entrega posterior de notificações (offline users)	Pass
WebSockets	
Notificação imediata de privilégios de administrador (online users)**	Pass
Página de administração atualizada em tempo real	Fail
Atualização imediata da lista de servidores multicast ativos**	Fail
REST	
Associar conta de utilizador ao Facebook	Pass
Partilha da página com o resultado de uma pesquisa no Facebook	Pass
Mostrar em cada resultado a língua original da página	Pass
Traduzir título e descrição das páginas para Português	Pass
Registo com a conta do Facebook (sem conta ucBusca)	Pass
Relatório	
Arquitetura do projeto Web detalhadamente descrita	Pass
Integração de Struts2 com o servidor RMI	Pass
Integração de WebSockets com Struts2 e RMI	Pass
Integração de REST WebServices no projeto	Pass
Extra (até 5 pontos)	
Utilização de HTTPS (4 pts)	Pass
Utilização em smartphone ou tablet (2pts)	Pass