



Sistemas Distribuídos

Relatório Meta 2

Membros do grupo:

Pedro Miranda
Rúben Alves

2012148969
2012142647

1. Introdução

A aplicação iVotas consiste numa plataforma de voto electrónico para as eleições do núcleo de estudantes e conselho geral da universidade de coimbra. Esta aplicação permite aos seus utilizadores votarem numa determinada lista de uma eleição desde que esta esteja a decorrer e pertença a universidade de coimbra.

2. Architecturas

A aplicação desenvolvida baseia-se numa arquitectura cliente-servidor construída anteriormente na meta 1. Esta arquitectura contém uma camada de persistência de dados composta por dois servidores RMI e uma base de dados MySQL. Os servidores RMI são responsáveis pela gestão do tráfego de dados entre outros componentes do projecto. Existem dois destes servidores, quando ativos irá ser estabelecida uma ligação entre eles, caso o um deles deixe de trabalhar este será substituído pelo outro. Para que isto aconteça os servidores RMI trocam mensagens UDP entre eles, sendo este processo chamado de "heartbeat".

O nosso sistema contém também diversos servidores multithread onde os clientes se poderão ligar via ligações TCP de modo a votarem na eleição pretendida. Cada servidor TCP estará associado a uma mesa de voto comunicando este com o servidor RMI de modo a permitir os utilizadores a votarem apenas nas eleições associadas a essa mesa.

Para esta meta foi desenvolvido um frontend web para a aplicação criada na primeira meta. De modo a tanto os utilizadores deste frontend, como os utilizadores com o software desktop instalado terem acesso aos mesmos dados, o frontend comunicará com o dataServer RMI, tal como os servidores multithread da aplicação.

A comunicação real-time de estatísticas entre servidor RMI e consola de administração foi implementada através de Websockets e de notificações. O servidor Web também deverá integrar, através de REST, a possibilidade de o utilizador se autenticar através do facebook. Esta aplicação segue uma arquitetura MVC.

A seguinte imagem representa a arquitectura seguida:

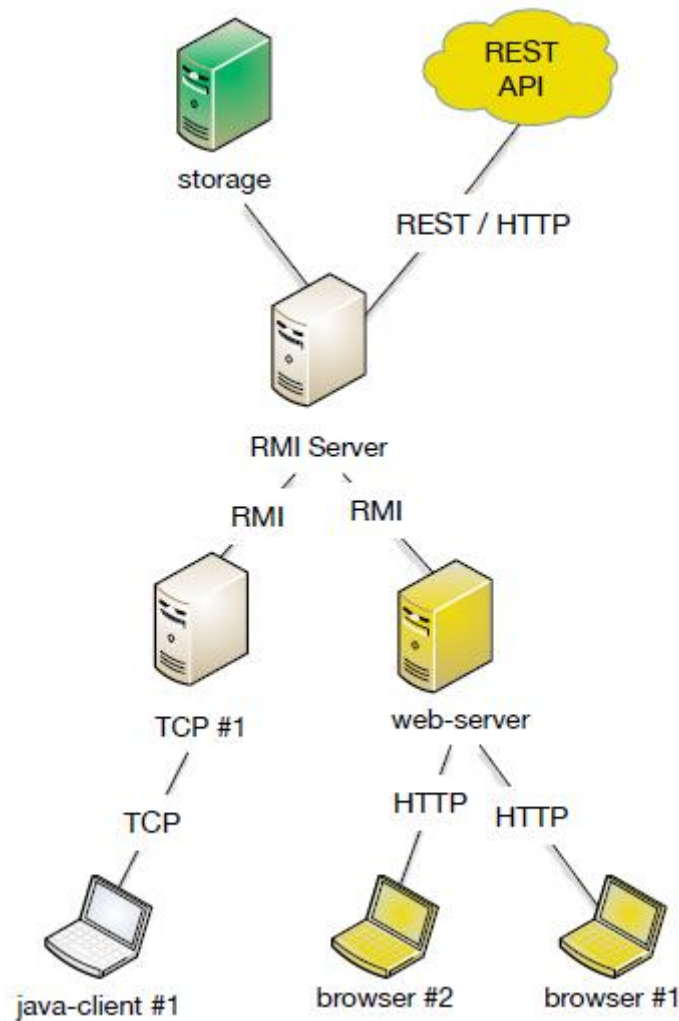


Figura 1: Arquitetura de Software

3. Modelo MVC

O modelo Model View Controller (MVC) é um modelo arquitetural muito usado no desenvolvimento de aplicações web. O controlador recebe todos os pedidos da aplicação, usando o modelo para preparar toda a informação necessária para a vista. Quando a vista obtém todos os dados do controlador gera a página visualizada pelo utilizador, o seguinte esquema representa o funcionamento desta norma:

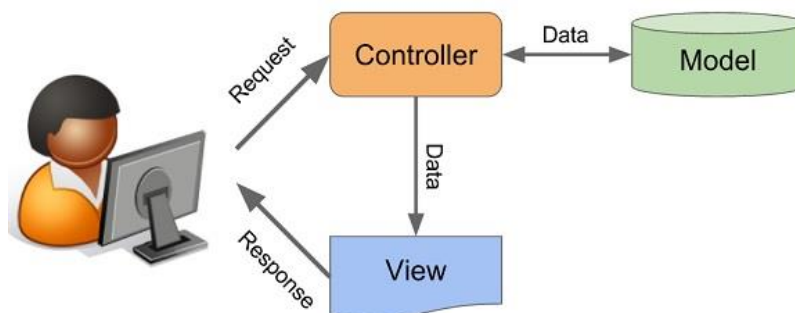


Figura 2: Diagrama do modelo MVC

4. Struts 2

O Struts2 é uma framework para o Java baseada no modelo MVC, onde são utilizados cinco componentes principais: acções (JavaBeans, POJOs), interceptors, Value Stack / OGNL, result e vistas (JSP). O modelo usado difere um pouco do MVC tradicional, uma vez que a acção assume o papel do modelo em vez do controlador. A figura abaixo representa a arquitectura implementada nesta framework:

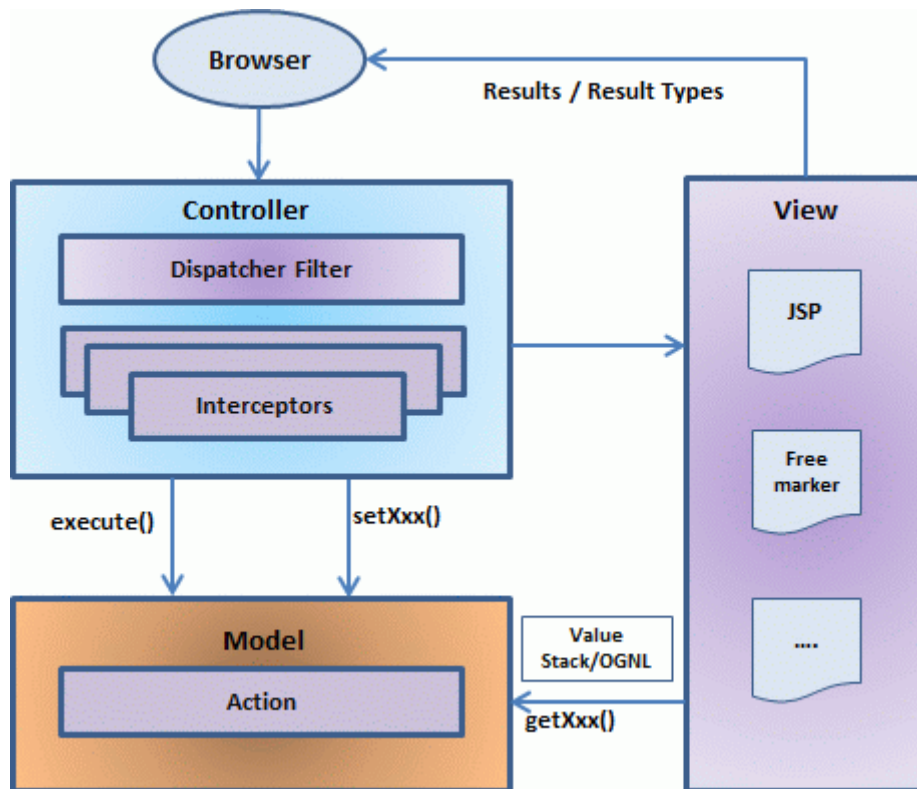


Figura 3: Diagrama do funcionamento do Struts 2

5. REST

No package data, está uma classe utilitária utilizada para obter informações para o Login através do Facebook, com recurso à biblioteca Scribe do Java. Através da API para o Facebook, é possível obter o URL de autenticação e a partir dele obter um token de acesso. Através deste token de acesso será possível fazer as acções no Facebook, como criar posts.

6. Manual da Arquitetura

O projecto é constituído por duas pastas principais: “SD_Project2” e “SD_Project2_TCP”.

A pasta “SD_Project2” encontra-se estruturada da seguinte forma:

- Contém o “SD_Project2.war” de modo a ser utilizado em qualquer servidor, para isso basta adicionar o ficheiro e executa-lo dentro do servidor.
- Interface gráfica e suas ações usando Struts 2: A pasta “WebContent” contém Java Server Pages (JSP) responsáveis por receber os dados inseridos pelo utilizador assim como apresentar o resultado das operações efectuadas. Operações estas que podem ser encontradas na pasta “/src/action”. O “struts.xml” é responsável pela ligação entre as JSPs e as operações que se pretende efectuar.
- Package “rmi_server” e “tcp_server”: Contêm uma versão actualizada do código do servidor RMI e do servidor TCP.

As seguintes imagens mostram a estrutura utilizada na organização do código fonte do projecto:

.settings	22/12/2017 14:58	Pasta de ficheiros	
build	22/12/2017 14:58	Pasta de ficheiros	
src	22/12/2017 17:11	Pasta de ficheiros	
WebContent	22/12/2017 14:58	Pasta de ficheiros	
.classpath	07/12/2017 17:21	Ficheiro CLASSPA...	1 KB
.project	27/11/2017 21:40	Ficheiro PROJECT	1 KB
SD_Project2.war	22/12/2017 14:56	Ficheiro WAR	4 527 KB

Figura 4: Pasta "SD_Project2"

bootstrap	META-INF	WEB-INF
college_create.jsp	college_create_feedback.jsp	college_delete_feedback.jsp
college_delete_list.jsp	college_edit.jsp	college_edit_feedback.jsp
college_edit_list.jsp	create_election.jsp	create_election_feedback.jsp
election_edit.jsp	election_edit_feedback.jsp	election_edit_lists.jsp
election_results.jsp	error.jsp	list_create.jsp
list_create_feedback.jsp	list_delete_feedback.jsp	list_delete_lists.jsp
list_edit.jsp	list_edit_feedback.jsp	list_edit_lists.jsp
NavBar.jsp	results_election_list.jsp	START.jsp
table_create.jsp	table_create_feedback.jsp	table_delete_feedback.jsp
table_delete_list.jsp	table_status.jsp	user_delete_feedback.jsp
user_delete_list.jsp	user_edit.jsp	user_edit_feedback.jsp
user_edit_list.jsp	user_register.jsp	user_register_feedback.jsp
user_votes.jsp	user_votes_list.jsp	

Figura 5: Pasta "WebContent"

action	22/12/2017 14:58	Pasta de ficheiros	
business	22/12/2017 14:58	Pasta de ficheiros	
data	22/12/2017 14:58	Pasta de ficheiros	
resources	22/12/2017 14:58	Pasta de ficheiros	
rmi_server	22/12/2017 14:58	Pasta de ficheiros	
tcp_server	22/12/2017 17:11	Pasta de ficheiros	
log4j.dtd	27/11/2017 18:19	Ficheiro DTD	7 KB
log4j.xml	27/11/2017 18:19	Documento XML	1 KB
struts.xml	21/12/2017 11:35	Documento XML	9 KB

Figura 6: Pasta "src"

A pasta “SD_Project2_TCP” contém uma estrutura semelhante á pasta “SD_Project2” com excepção dos packages “rmi_server” e “tcp_server”.









	.settings	22/12/2017 17:11	Pasta de ficheiros	
	build	22/12/2017 17:11	Pasta de ficheiros	
	rest	22/12/2017 17:11	Pasta de ficheiros	
	src	22/12/2017 17:11	Pasta de ficheiros	
	WebContent	22/12/2017 17:11	Pasta de ficheiros	
	.classpath	21/12/2017 22:36	Ficheiro CLASSPA...	1 KB
	.project	21/12/2017 15:39	Ficheiro PROJECT	1 KB
	SD_project.war	22/12/2017 17:10	Ficheiro WAR	5 332 KB

Figura 7: Pasta "SD_Project2_TCP"










	bootstrap	22/12/2017 17:11	Pasta de ficheiros	
	META-INF	22/12/2017 17:11	Pasta de ficheiros	
	WEB-INF	22/12/2017 17:11	Pasta de ficheiros	
	error.jsp	18/12/2017 17:14	Ficheiro JSP	1 KB
	login.jsp	22/12/2017 15:32	Ficheiro JSP	4 KB
	user_login.jsp	22/12/2017 00:38	Ficheiro JSP	3 KB
	vote.jsp	21/12/2017 00:52	Ficheiro JSP	1 KB
	vote_Election.jsp	21/12/2017 00:47	Ficheiro JSP	2 KB
	vote_lists.jsp	21/12/2017 03:59	Ficheiro JSP	2 KB

Figura 8: Pasta "WebContent"







	action	22/12/2017 17:11	Pasta de ficheiros	
	data	22/12/2017 17:11	Pasta de ficheiros	
	rest	22/12/2017 17:11	Pasta de ficheiros	
	log4j.dtd	27/11/2017 18:19	Ficheiro DTD	7 KB
	log4j.xml	27/11/2017 18:19	Documento XML	1 KB
	struts.xml	22/12/2017 15:08	Documento XML	2 KB

Figura 9: Pasta "src"

7. Check List

100		Nota Final:	100	74
55	Requisitos Funcionais		55	60
5	Registar pessoas (estudantes, docentes, ou funcionários)		5	5
5	Login protegido com password (acesso a ações e a páginas)		5	5
5	Criar eleição (incl. integração com a meta 1)		5	5
5	Criar listas de candidatos a uma eleição		5	5
5	Listar eleições e consultar detalhes de cada uma delas		5	5
5	Adicionar mesas de voto a uma eleição (incl. integração com a meta 1)		5	5
5	Alterar propriedades de uma eleição		5	5
5	Votar (incl. integração com a meta 1)		5	5
5	Saber em que local votou cada eleitor		5	5
5	Eleição termina corretamente na data, hora e minuto marcados		5	5
5	Consultar resultados detalhados de eleições passadas		5	5
0	Grupos de 3: Alterar dados pessoais (-5)		0	5
0	Grupos de 3: Gerir membros de cada mesa de voto (-5)		0	
0	Grupos de 3: Considerar eleições de departamento e faculdade (-5)		0	
15	WebSockets		15	0
5	Página de uma eleição mostra eleitores em tempo real		5	
5	Páginas de administração mostram o estado das mesas de voto (da meta 1)		5	
5	Listar utilizadores online		5	
20	REST		20	7
5	Associar conta existente ao Facebook		5	
5	Login com o Facebook		5	5
5	Partilha da página de uma eleição no Facebook sendo o post atualizado no f		5	2
5	Post no Facebook de um eleitor assim que vote numa eleição		5	
0	Grupos de 3: deve ser possível desassociar um utilizador do facebook (-5)		0	
10	Relatório		10	7
2	Arquitetura do projecto Web detalhadamente descrita		2	2
2	Integração do Struts com o servidor RMI		2	1
2	Integração de WebSockets com Struts e RMI		2	1
2	Integração de APIs REST no projecto		2	1
2	Testes de software (tabela: descrição e pass/fail de cada teste)		2	2
	Extra (até 5 pontos)		0	0
	Utilização de HTTPS (4 pts)			
	Utilização em smartphone ou tablet (2pts)			