



UNIVERSIDADE D  
**COIMBRA**

**Faculdade de Ciências e Tecnologias da Universidade de Coimbra**

## **Motor de Busca - Relatório**

**Unidade Curricular: Sistemas Distribuídos**

**Docente: Hugo Dinis Pereirinha da Silva Amaro**

**Henrique José Correia Brás, 2021229812**

**Tiago Rafael Cardoso Santos, 2021229679**

# Índice

<b>Introdução</b>	<b>2</b>
<b>Arquitetura</b>	<b>2</b>
<b>Multicast(Meta1)</b>	<b>3</b>
<b>RMI (Meta1)</b>	<b>4</b>
<b>SpringBoot</b>	<b>4</b>
<b>WebSockets</b>	<b>4</b>
<b>REST API</b>	<b>4</b>
<b>Distribuição de Tarefas (Meta1)</b>	<b>4</b>
<b>Testes de Software</b>	<b>5</b>
<b>Conclusão</b>	<b>5</b>

# Introdução

Este trabalho prático tem como objetivo desenvolver um motor de busca de páginas Web, similar ao Google. Na primeira meta foi criada a parte de Backend, onde o sistema inclui indexação automática, busca de informações, ordenação de resultados por relevância e indexação de novos urls. Na segunda meta foi criada a parte de Frontend, com a utilização de SpringBoot e WebSockets.

## Arquitetura

A arquitetura utilizada, disponibilizada no projeto pelos docentes da unidade curricular(figura 1), descreve a forma que os Downloaders, Index Storage Barrels, Gateway e Client comunicam.

O Cliente comunica com a gateway por RMI, e vice-versa, escolhendo uma de quatro possibilidades, indexar novo URL, pesquisar URLS por palavras, ver a página admin em tempo real e pesquisar URLS.

A Gateway, recebendo URLS dos clientes, adiciona-os a uma lista e envia um de cada vez para os Downloaders, até um destes estar disponível, por RMI.

Os Downloaders, recebendo o URL da gateway, geram todos os links conectados a este, e é feita a recursão de cada link descoberto, enviando os resultados para a Gateway e recebendo um novo URL, por RMI. À medida que novos links são gerados, estes são também enviados e guardados nos Index Storage Barrels, assim como os termos associados a cada um, por Multicast.

Os Index Storage Barrels armazenam os links enviados pelos downloaders, e enviam para a gateway os resultados esperados por RMI.

Com estas ligações formadas, a parte de Backend ficou finalizada na meta anterior onde só tivemos que alterar os returns de cada função e procedemos à utilização de SpringBoot para que a Gateway conseguisse comunicar com o Web Server, por RMI. A Rest API utilizada é interna, tratando assim dos pedidos dos clientes.

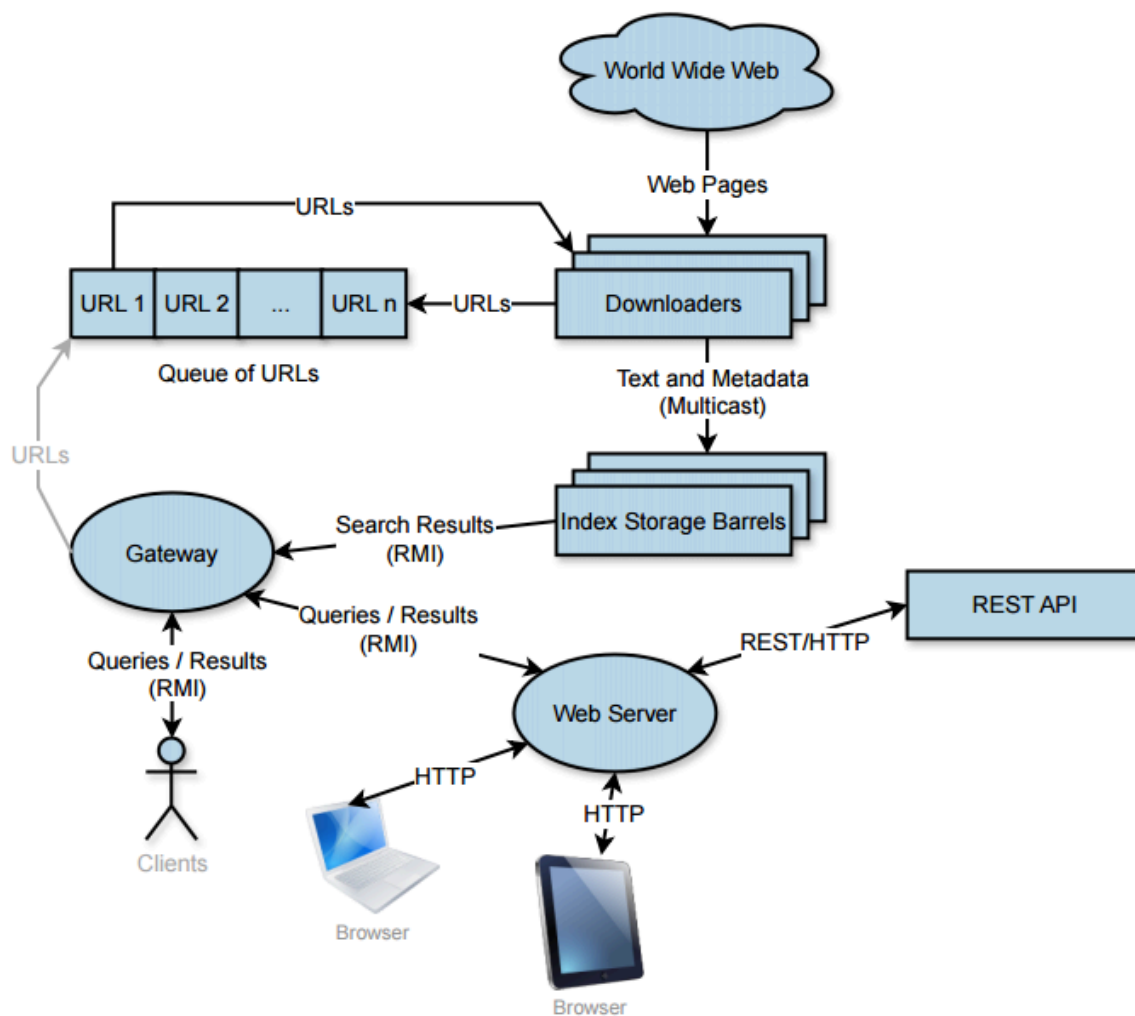


Figura 1

## Multicast(Meta1)

A comunicação por Multicast tem como objetivo enviar a mesma informação para diversos recetores simultaneamente, utilizando apenas uma única mensagem.

Neste projeto, é possível observar que os Downloaders comunicam por Multicast com os Index Storage Barrels, para que os urls sejam enviados para um ou mais Barrels disponíveis, mantendo a sincronia entre estes, para que armazenem informações idênticas.

Os Barrels criam uma socket via UDP, conectam-se à porta dos Downloaders, juntam-se ao grupo de Multicast e recebem então a informação enviada, fechando depois a socket quando as tarefas se encontram todas finalizadas.

Os Downloaders criam também uma socket via UDP, indicam o IP que será utilizado, enviam a informação para os Barrels e fecham a socket após as tarefas estarem todas finalizadas.

## RMI (Meta1)

A comunicação por RMI tem como objetivo permitir a aplicações Java invocar métodos e funções de outras aplicações Java.

Neste projeto, é possível observar que o Cliente, a Gateway e os Downloaders comunicam por RMI, e os Barrels comunicam com a Gateway também por RMI. Desta forma, estas aplicações conseguem invocar funções de outras, fazendo assim uma comunicação direta e simplificada, como se tivessem a ser chamadas localmente.

## SpringBoot

O Springboot facilita o desenvolvimento de aplicações web configurando automaticamente a aplicação, suportando também a criação rápida de uma REST API. Desta forma, é possível realizar a ligação entre o código de backend e frontend.

A ligação através de RMI com a aplicação é feita através da criação de um bean que registra o objeto de implementação.

## WebSockets

Os WebSockets realizam a comunicação com o cliente por HTTP e com a gateway por RMI, permitindo uma comunicação bidirecional de informações, sendo muito úteis em projetos deste género, que necessitam de interações atualizadas em tempo real.

A conexão do WebSocket com o SpringBoot é feita através da criação de um Controller, que inclui métodos para processar estas mensagens recebidas.

A ligação RMI é feita dentro do manipulador de mensagens WebSocket, que envia de volta ao cliente a resposta pedida.

## REST API

A REST API permite ao cliente comunicar com o servidor via HTTP com os pedidos de GET e POST.

Quando o cliente quer saber o top 10 de pesquisas, é utilizado o método GET para obter essas informações, para as outras funcionalidades, é usado o método POST para enviar o pedido.

## Distribuição de Tarefas (Meta1)

Assim como foi sugerido, a distribuição de tarefas foi feita com o aluno Tiago Cardoso ficando responsável pelos Downloaders e pela componente multicast dos Barrels, e o aluno Henrique Brás ficando responsável pela Gateway e pela componente RMI dos Barrels.

Como era esperado, ambos os alunos tiveram um papel crucial em todas as funcionalidades












desenvolvidas no decorrer do projeto, auxiliando no que era necessário e tendo um conhecimento geral em tudo o que foi implementado, fazendo assim um pouco de cada.

Quanto ao cliente, o trabalho foi feito de maneira simultânea, e à medida que o projeto ia sendo finalizado, o aluno Henrique Brás passou a ficar responsável pelo relatório e ficheiro JavaDoc, enquanto que o aluno Tiago Cardoso implementava os últimos detalhes precisos no código.

Desta forma, o trabalho foi bem distribuído e balanceado para que cada aluno tivesse um número similar de responsabilidades e tarefas.

## Testes de Software

Como teste, decidimos usar o link <http://www.uc.pt>, e obtivemos os seguintes resultados:

- Indexar URL: Enviamos o URL -> Resposta: Link enviado com sucesso 
- Procurar palavra: Escolhemos uma palavra que se encontrava no link, e outra que não.  
Palavra não obtida(Amarante) -> Resposta: Nao existem ocorrencias de Amarante   
Palavra obtida(sobre) -> Resposta -> 38 links 
- Modo Admin:  
Resposta -> Número de downloaders ativos: 1   
Resposta -> Número de links existentes: 168 
- Pesquisar ligação de link: Pesquisamos o URL.  
Resposta -> Links conectados   
(<http://www.sd.pt>) -> Resposta -> Nao existem ocorrencias do link <http://www.sd.pt> 
- Top 10 pesquisas:  
Resposta -> sobre 1 ocorrencia   
coimbra 1 ocorrencia   
Amarante 1 ocorrencia   
<http://www.sd.pt> 1 ocorrencia 

## Conclusão

Em suma, a integração eficaz de RMI e Multicast facilitou uma comunicação fluida entre os componentes do sistema, enquanto que os WebSockets proporcionaram atualizações em tempo real essenciais para a funcionalidade de administração. Através deste projeto, conseguimos adquirir habilidades práticas em desenvolvimento web e sistemas distribuídos.