



UNIVERSIDADE D
COIMBRA

Googol: Motor de pesquisa de páginas Web

Sistemas Distribuídos


Licenciatura em Engenharia Informática
2022/2023

30 de março de 2024

Autores


João Moreira

 joaomoreira@student.dei.uc.pt

 2020230563

Tomás Pinto

 tomaspinto@student.dei.uc.pt

 2020224069

Índice

1	Introdução	3
2	Arquitetura	3
2.1	Downloaders	4
2.1.1	Multicast	4
2.2	Index Storage Barrels	5
2.3	RMI Search Module	5
2.3.1	RMI e RMI callbacks	5
2.4	RMI Client	7
3	Failover	7
4	Testes	8
5	Distribuição de Tarefas	9

1. Introdução

Neste projeto foi-nos pedido que desenvolvêssemos um motor de pesquisa de paginas Web que reunisse um conjunto de funcionalidades parecidas a serviços conhecidos do dia a dia (Google, Bing, Duckduckgo).

Algumas das mais importantes funcionalidades são a indexação automática (Web crawling), onde os utilizadores podem especificar URLs para serem indexados pelo sistema, e a partir destes urls indexar recursivamente possíveis ligações encontradas em cada página, e também a busca (search engine) onde são obtidas a lista de páginas que possuam as palavras pesquisadas.

Com a realização deste trabalho tivemos como objetivo desenvolver as nossas capacidades em alguns temas como: arquitetura cliente-servidor, comunicação multicast, modelos multithread de servidores e Java RMI.

2. Arquitetura

Foi-nos especificada uma arquitetura global para o projeto bem como uma explicação do que iriam consistir os vários programas que teríamos de desenvolver. Na Figura 1 podemos ver uma representação da arquitetura global do projeto que iremos descrever detalhadamente de seguida.

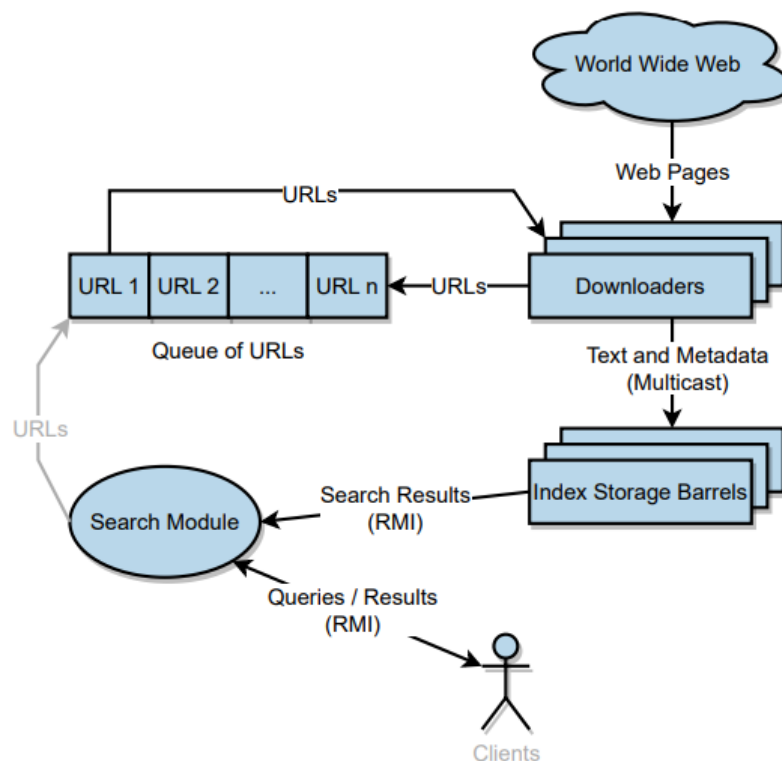


Figura 1: Estrutura do projeto

A aplicação consiste em quatro programas: Downloaders, Index Storage Barrels, RMI Search Module e RMI Client.

2.1. Downloaders

Os *Downloaders* são compostos por diferentes *threads* a trabalhar em paralelo no mesmo servidor. O número de *threads* a lançar pode ser pré-definido pelo programador.

Cada *thread* dá *handle* de um objeto *Reader*, cujos ficaram armazenadas num *ArrayList* no programa principal. Esta decisão ajuda-nos a ter controlo do/das estado/estatísticas dos *Downloaders*, onde iremos falar mais à frente.

A *queue* utilizada é uma *Blocking Queue* implementada por nós, com duas funcionalidades muito simples:

1. *pop* - devolve e remove o elemento da *head*.
No caso de estar vazia, a *thread* fica em espera bloqueante.
No caso de estar cheia¹, o elemento não é adicionado.
2. *add* - coloca um elemento na fila.
No caso de inicialmente a fila estar vazia, as *threads* em espera serão notificadas que já existe *URLs* para processar.

Para facilitar a comunicação entre *Downloaders* e *Queue de URLs*, esta estrutura está armazenada no servidor principal, logo não é um programa à parte, tal como foi dada a possibilidade. Para além disso, o servidor principal conta com um *ArraySet* para armazenar os *URLs* já processados, para que não exista trabalho repetido.

O processamento dos *URL* será feito com ajuda da biblioteca *Jsoup*, onde falaremos num ficheiro à parte como poderá ser feita a instalação.

2.1.1. Multicast

O envio da informação processada para os *Storage Barrels* é feita por *multicast*. Para tal criamos um protocolo muito simples: enviamos uma palavra-chave, o *URL* correspondente e o valor da palavra-chave, separado por espaços.

Palavras-chave:

- Title - Envia o titulo encontrado no documento HTML de um certo URL;
- Token - Envia uma palavra encontrada no documento HTML de um certo URL;
- Url - Envia um URL resultante da indexação;
- Paragraph - Envia um paragrafo encontrada no documento HTML de um certo URL;

Palavra-Chave	URL	Valor
Title	wikipedia.org/wiki/Universidade_de_Coimbra	Universidade de Coimbra
Token	wikipedia.org/wiki/Universidade_de_Coimbra	Universidade
Token	wikipedia.org/wiki/Universidade_de_Coimbra	de
Token	wikipedia.org/wiki/Universidade_de_Coimbra	Coimbra
Url	wikipedia.org/wiki/Universidade_de_Coimbra	wikipedia.org/wiki/Coimbra
Paragraph	wikipedia.org/wiki/Universidade_de_Coimbra	A Universidade de Coimbra ...

Tabela 1: Exemplo do Protocolo de Comunicação de Multicast

¹O tamanho da fila pode ser definida previamente pelo programador no momento da sua declaração.

2.2. Index Storage Barrels

No caso do *Index Storage Barrels*, ao contrário dos *Downloaders* são programas independentes, ou seja, para ter diversos *Barrels* a trabalhar em paralelo, o administrador terá de correr o programa tantas vezes quantos *Barrels* queira.

Neste "servidor" para evitar perdas de informação, no caso de existir falhas, cada *Barrel* contém um ficheiro próprio, onde lê e escreve. Por este motivo, quando o programa é executado tem de se passar o nome do ficheiro por parâmetro, este será lido e a informação contida vai ser extraída para as estruturas de dados.

A informação recebida por *multicast* por parte dos *Downloaders* é guardada no ficheiro exatamente sem alterações. Deste modo, permite-nos fazer o *parsing* da mesma forma, tanto pelo que recebemos do *multicast* como o que está no ficheiro.

Nos ficheiros também está contida informação sobre as estatísticas de pesquisas realizadas (é guardada a pesquisa e o número de vezes que foi feita) e também registo de utilizadores (*username* e *password*).

Palavra-Chave	Valor1	Valor2
Register	User123	Pass1234
Search	Como fazer um Googol	500

Tabela 2: Pesquisas e registos dos utilizadores presentes no ficheiro

Como os *Barrels* vão ser responsáveis por armazenar todos os dados necessários para as várias operações/funcionalidades da aplicação, temos como estruturas as seguintes:

- *invertedIndex* - guarda todos os *URLs* que contém determinado *token*.
- *relevanteIndex* - guarda todos os *URLs* que apontam para um determinado *URL*.
- *titles* - guarda o título de um determinado *URL*.
- *Paragraph* - guarda um paragrafo de um determinado *URL*.
- *searches* - guarda o número total de vezes que foi realizada uma certa pesquisa .
- *Regists* - guarda os registos de utilizadores.

2.3. RMI Search Module

O RMI search module é importante como intermediador entre os pedidos efetuados pelos RMI clients e os dados necessários para satisfazer esses pedidos que se encontram nos Storage Barrels.

2.3.1. RMI e RMI callbacks

O search module vai ser um servidor na comunicação com os RMI clients via RMI, possuindo os métodos remotos que são utilizados por estes para satisfazer as diversas funcionalidades dos utilizadores.

Vai ser feita a comunicação com os Storage Barrels via RMI também, utilizando RMI callbacks, fazendo com que ambos os programas utilizem métodos remotos um do outro, o que é necessário uma vez que queremos fazer operações acedendo aos dados existentes nos Barrels. Isto

é possível pois cada Barrel ao iniciar conecta-se ao Search module passando o objeto de callback (instancia de objeto criada no Barrel), que pode ser utilizado por este para comunicar de volta com o Barrel específico. O Search module vai ter o objeto de callback de todos os Barrels ativos no sistema podendo controlar a qual acede em específico.

De modo a passar os URLs introduzidos pelos utilizadores (recebidos desde os RMI Clients) até ao Servidor dos Downloaders, é feita a comunicação do mesmo com o Search module via RMI também, onde o Search module irá invocar um método remoto para passar o URL para o servidor. O mesmo acontece para obter no Search Module o numero de downloaders ativos que contribui para as estatísticas do sistema.

Métodos remotos do Search Module:

- sayURL - Método chamado no RMI Client com objetivo de obter no Search Module o URL escolhido a ser indexado, para ser posteriormente enviado ao servidor dos Downloaders.
- saySearch - Método chamado no RMI Client com objetivo de obter no Search Module a busca feita pelo utilizador, para ser posteriormente enviada aos Barrels.
- pointToLink - Método chamado no RMI Client com objetivo de obter no Search Module o URL escolhido para obter as ligações conhecidas que apontem para essa página, para ser posteriormente enviada ao Barrel.
- Register - Método chamado no RMI Client que vai receber o username e password para efetuar o registo de um utilizador.
- Login - Método chamado no RMI Client que vai receber o username e password para efetuar o login de um utilizador.
- Stats - Método chamado no RMI Client que vai recolher as estatísticas (nº de Downloaders e Barrels) e também a lista das pesquisas mais feitas e apresentar ao utilizador.
- registerBarrel - Método chamado nos Barrels ao iniciarem com objetivo destes serem registados no Search module para as comunicações futuras.
- logoutBarrel - Método chamado nos Barrels quando é detetado um SIGINT ou exception que leve ao término do Barrel, com objetivo destes serem retirados do registo de Barrels ativos no Search module.

Métodos remotos do server de Downloaders:

- GetURL - Método utilizado no Search module com objetivo de enviar o URL inserido pelo utilizador para indexação no servidor de Downloaders.
- getNdownloaders - Método utilizado na função stats do Search module com objetivo de obter o numero de Downloaders a trabalhar, para ser posteriormente enviado ao utilizador.

Métodos remotos dos Storage Barrels (Callbacks):

- getSearch - Método chamado no Search Module com objetivo de enviar a busca feita para ser processada nos barrels.
- getPointToLink - Método chamado no Search Module com objetivo de enviar o url introduzido pelo utilizador para ser processado nos barrels.
- GetInfos - Método chamado no Search Module com objetivo de obter a lista das dez pesquisas mais feitas por utilizadores.

- Regist - Método chamado no Search Module com objetivo de enviar o username e password introduzidos pelo utilizador para serem processados nos Barrels e realizar registo.
- Login - Método chamado no Search Module com objetivo de enviar o username e password introduzidos pelo utilizador para serem processados nos Barrels e realizar login.

2.4. RMI Client

O RMI Client é o cliente que os utilizadores vão usar para poder aceder às funcionalidades da aplicação. Vai comunicar com o servidor Search Module via RMI e assim invocar os metodos remotos deste.

Implementamos uma interface simples com as várias funcionalidades a que o utilizador pode aceder, sendo estas:

- Fazer indexação de um URL;
- Realizar uma pesquisa;
- Fazer registo na aplicação;
- Fazer login na aplicação;
- Obter estatísticas do sistema (nº de Downloaders e Barrels);
- Obter as dez pesquisas mais feitas por utilizadores;
- Obter todas as ligações conhecidas que apontem para uma página (caso o utilizador tenha registo e login feitos);
- Opção de sair;

3. Failover

Para evitar erros de *Failover*, existe a necessidade de quebrar a ligação RMI entre o *Search Model* e o *Index Storage Barrel* quando existe uma falha.

Sendo assim, quando *Barrel* dá *handle* de exceções, ele avisa o *Search Model* que irá desligar-se por RMI. Deste modo, o *Search Model* remove a ligação e remove-o da lista de *Barrels* ativos, fazendo assim com que este *Barrel* não seja mais escolhido para performar tarefas e seja escolhido outro ativo.

4. Testes

Na secção seguinte enumeramos os testes feitos à aplicação. Em cada teste especificamos o procedimento que o utilizador terá de seguir para realizar a funcionalidade, bem como os casos de sucesso e insucesso que podem advir da mesma. Colocamos para cada "Aceite" caso a nossa aplicação cubra tal comportamento e "Rejeitado" caso contrário.

Indexar novo URL		
Procedimento	Escolher a opção "Index URL" do Menu dos utilizadores e introduzir o URL.	
Sucesso	O utilizador introduz um URL válido que irá ser indexado.	Aceite
Sucesso	O programa indexa URLs recursivamente.	Aceite
Insucesso	O utilizador introduz um URL inválido (tudo o que não seja um URL).	Aceite

Tabela 3: Testes de Indexar URLs

Pesquisar páginas que contenham um conjunto de termos		
Procedimento	Escolher a opção "Search" do Menu dos utilizadores e introduzir os termos de pesquisa.	
Sucesso	O utilizador introduz termos que contém todos os termos da pesquisa (título, URL e citação).	Aceite
Sucesso	As páginas aparecem ordenadas por número decrescente de ligações apontadas para a própria	Aceite
Sucesso	Os resultados da pesquisa são disponibilizados 10 a 10	Rejeitado
Insucesso	Nenhuma página contém todos os termos da pesquisa.	Aceite

Tabela 4: Testes de pesquisa

Registar-se (Sign up)		
Procedimento	Escolher a opção "Register" do Menu dos utilizadores e introduzir os dados (username e password)	Aceite
Sucesso	O utilizador introduz valores de username e password novos a ser registados.	Aceite
Insucesso	O utilizador introduz um username que já se encontra registado.	Aceite

Tabela 5: Teste de registo

Fazer login		
Procedimento	Escolher a opção "Log in" do Menu dos utilizadores e introduzir os dados (username e password)	Aceite
Sucesso	O utilizador introduz valores de username e password correspondentes a uma conta registada.	Aceite
Insucesso	O utilizador introduz um username que não está registado.	Aceite
Insucesso	O utilizador introduz a password errada para o username introduzido.	Aceite

Tabela 6: Teste de login

Consultar lista de páginas com ligação para uma página específica		
Procedimento	Ter registo e login feitos. Escolher a nova opção “Get pointed links” do Menu dos utilizadores e introduzir o URL da página.	Aceite
Sucesso	O utilizador uma página que possui ligações para a que introduziu.	Aceite
Insucesso	Nenhuma página possui ligação para a introduzida.	Aceite

Tabela 7: Testes de consulta de ligações

Consultar informações sobre o sistema e 10 pesquisas mais feitas por utilizadores.		
Procedimento	Escolher a nova opção “Get Stats” do Menu dos utilizadores.	Aceite
Sucesso	O utilizador recebe informação sobre o numero de Downloaders e Barrels ativos e também as 10 pesquisas mais feitas por utilizadores.	Aceite
Sucesso	O utilizador recebe listas de informação sobre os Downloaders e Barrels ativos (porto e ip).	Rejeitado

Tabela 8: Testes de consulta de estatísticas

5. Distribuição de Tarefas

Relativamente à distribuição de tarefas no grupo, decidimos seguir a sugestão proposta no enunciado do projeto.

As tarefas ficaram distribuídas da seguinte forma:

- João Moreira - Responsável pelos Downloaders e pela componente multicast dos Barrels.
- Tomás Pinto - Responsável pelo Search Module e pela componente RMI dos Barrels

Apesar de cada um se ter focado mais na sua parte, as decisões tomadas e implementações feitas foram discutidas mutuamente ao longo da realização do trabalho. Em alguns momentos os dois acabamos por trabalhar em ambas as partes.

O relatório e os testes à aplicação foram realizados por ambos os elementos em conjunto.