Universidade do Minho

Mestrado Integrado em Engenharia Informática

Laboratórios de Informática III

Relatório - Stack Overflow

Grupo 9:
Carlos Gomes A77185
Nuno Silva A78156

9 Junho 2018

Conteúdo

Introdução

Este projeto foi nos solicitado pelos docentes da unidade curricular de Laboratórios de Informática III e tem como principal objetivo a realização de um programa que permita analisar os posts presentes em backups do Stack Overflow, realizados em diferentes datas, e extrair informação útil para esse período de tempo como, por exemplo, os top N utilizadores com maior número de posts de sempre, obter o número total de posts, entre outros.

Outros objetivos estão também associados a este, como a consolidação de conhecimentos adquiridos em unidades curriculares anteriores, dentro das quais se incluem Programação Imperativa, Algoritmos e Complexidade e Arquitetura de Computadores.

A semântica deste trabalho segue a mesma do trabalho realizado numa primeira fase na Linguagem C, agora vai ser implementado na Linguagem Java. Este projeto envolve uma enorme quantidade de dados e uma complexidade estrutural e algorítmica elevada. O principal desafio do projeto seria a programação em larga escala, uma vez que pelo nosso programa passam milhares de dados, aumentando assim a complexidade do trabalho. Para que a realização deste projeto fosse possível, foram-nos introduzidos novos princípios de programação, com especial relevo para a Modularidade e encapsulamento de dados.

Estrutura Adotada

Enquanto equipa, depois de termos feito o último projeto em C, onde utilizamos lista ligadas, decidimos mudar a nossa estrutura. Para isso, começamos a pesquisar no sistema de procura, o tão famoso Google, a melhor estrutura em termos de velocidade a carregar para estrutura e ler a estrutura. Com isto, enquanto equipa, ficamos entre HashMaps e TreeSets. No entanto, optamos por HashMap, já que para carregar é mais rápido que os TreeSets e também tem boa velocidade a ler da estrutura, já as TreeSets demora um bocado mais a carregar.

Ora, a nossa estrutura principal vai consistir em quatro HashMaps, cada um vai guardar a informação que precisamos para resolver as questões propostas pela equipa docente.

```
public class TCDExample implements TADCommunity {

    private MyLog qelog;
    HashMap<Long, Posts> hMapPosts;
    HashMap<Long, Usuarios> hMapUsers;
    HashMap<Long, Comentarios> hMapComments;
    HashMap<Long, Tags> hMapTags;
}
```

Em cada HashMap, vai ter as suas próprias variáveis necessárias para a realização do projeto.

Classes MyHandler

As classes MyHandler(MyHandlerPosts, MyHandlerUsers, MyHandlerComments, MyHandlerTags) tem como o objetivo de guardar toda a informação num objeto da classe onde se encontram as variáveis de cada ficheiro com extensão .xml nos HashMaps apropriados, onde a key para ter a acesso a esses objetos ser o ID.

Estas classes têm dois métodos fundamentais, o startElement e o endElement. O método startElement tem como objetivo a leitura e a atribuição dos valores as variáveis de cada objeto, contudo o método endElement lê os ficheiros xml e inicia novos objetos por cada row encontrada.

```
public class MyHandlerPosts extends DefaultHandler {
    //Hash para guardar o objeto de Posts
    private HashMap<Long, Posts> hMapPosts;
    Posts post = new Posts();

    public MyHandlerPosts(){
        hMapPosts = new HashMap<>();
    }

    //metodo get para a lista de posts
    public HashMap<Long, Posts> getHashMapPosts(){
        return hMapPosts;
    }

    public void startElement (String uri, String localName,
        String qName, Attributes attributes) throws SAXException{...}

    public void endElement(String uri, String localName,
    String qName) throws SAXException{...}
}
```

Parser

À semelhança da primeira fase do projeto, no dump fornecido encontravam-se ficheiros no formato XML, sendo por isso necessário haver a passagem para as estruturas em questão.

Como tal, foi usado o package javax.xml.parsers.ParserConfigurationException, javax.xml.parsers.SAXParser, javax.xml.parsers.SAXParserFactory, org.xml.sax.Attributes, org.xml.sax.helpers.DefaultHandler, org.xml.sax.SAXException, para que fosse possível efetuarmos o parsing.

Após várias pesquisas descobri-mos o SAXParser e achamos adequado ao nosso projeto pois o parser opera em cada parte do documento XML sequencialmente, emitindo eventos de análise enquanto fazem a passagem única pelo fluxo de entrada.

De referir que dentro da classe Parser temos definidos quatro métodos cada um deles encarregue de fazer o parse ao ficheiro a que lhe foi atribuído.

Conclusão

Após a conclusão desta segunda e última fase de desenvolvimento do projeto, podemos afirmar que, nesta fase em particular, a principal dificuldade encontrada foi o facto do volume de dados analisado ser bastante elevado, daí que tenha sido tido um cuidado redobrado com a forma como o programa foi estruturado, de forma a que permitisse uma maior rapidez em run time.

Após uma análise global sobre o projeto, acreditamos veemente que desenvolvemos uma boa solução, sendo essa capaz de processar o volume de dados apresentado e produzir respostas a interrogações que lhe são colocadas num tempo útil.

Por fim, acrescenta-se também que este projeto foi positivo na medida em que nos permitiu implementar e aprofundar o conhecimento de algorítmos/estruturas abordados noutras unidades curriculares, assim como desenvolver a nossa capacidade de programação numa linguagem orientada a objetos que é o Java.