Programação Imperativa (3º ano de Curso)

Trabalho Prático Nº 1

Relatório de Desenvolvimento

Ricardo Ferreira (A82568)
Diogo Rocha (A79751)
Tiago Sousa (A81922)

5 de Abril de 2020

Resumo

No presente relatório é apresentada a resolução de um exercício referente ao TP1, que tem como objectivos a utilização de Expressões Regulares para descrição de padrões de frases,identificar as acções semânticas a realizar como reacção ao reconhecimento de cada um desses padrões,identificar estruturas de dados que possam vir a ser necessárias para armazenar temporariamente a informação extraída do texto fonte e a utilização do Flex para gerar filtros de texto em C.

Conteúdo

Capítulo 1

Introdução

O grupo analisou os enunciados disponíveis e escolheu o **4 - Transformador Publico2NetLang**. Este enunciado apresenta-nos um tema relacionado com vários comentários de vários utilizadores a uma noticia publicada no jornal *O Público*, extraídos da pagina HTML da versão online do dito jornal.

Ao longo deste trabalho produzimos essencialmente um filtro que lê o ficheiro HTML, verificando se está perante um Comentário ou a uma lista de respostas a esse Comentário e carrega a informação do ficheiro numa estrutura pensada ao pormenor para adquirir toda a informação presente num Comentário. Após o ficheiro não ter mais Comentários, e assim a sua estrutura estar totalmente carregada, é possível obter a informação através do output do terminal ou através de um ficheiro em formato json.

Com este relatório pretendemos apresentar as nossas opções, algoritmos desenvolvidos e ainda a estrutura utilizada para a realização do filtro. Pretendemos também apoiar aquelas que foram as nossas soluções, com conhecimento obtido nas aulas teóricas, relembrando por exemplo o caso dos autómatos.

Para uma melhor visão do que irá ser abordado neste relatório deixamos uma breve descrição daquilo que foi feito. No segundo capítulo foi feita uma análise informal e uma especificação dos requisitos deste projecto. No terceiro capítulo foi realizado o desenho da concepção no qual estão envolvidos os algoritmos e estruturas de dados usados. No quarto capítulo mostramos alguns exemplos de implementações e vários resultados de testes realizados. Por último no capítulo 5 fazemos uma retrospectiva do trabalho realizado e concluímos.

Capítulo 2

Análise e especificação

Analisando o problema como um todo o que podemos encontrar aquando da observação do ficheiro *Publico_extraction_portuguese_comments_4.html* é um ficheiro bem organizado com uma estrutura conhecida em que reconhece quando existe uma resposta ou mais a um comentário.

Nas secções seguintes será apresentado o objectivo do exercício e ainda as observações que foram feitas ao ficheiro que contém todo o HTML, por forma a pensar que casos iríamos ter futuramente e começarmos a delinear uma arquitectura duma possível solução.

2.1 Análise e especificação dos requisitos

2.1.1 Lista de Comentários

No exercício era requerido um filtro que extraísse o conteúdo relevante dos comentários de forma a fazer um estudo sócio-linguístico de forma e conteúdo destes.

Ao proceder à análise do ficheiro HTML reparamos que cada comentário teria a seguinte estrutura, respectivamente:

- Nome do Autor
- Data do Comentário
- Conteúdo do Comentário
- Respostas ao Comentário(caso existam)

Posto isto, comparando com a estrutura do enunciado passamos a ter de tratar menos um elemento, pois este não está presente na estrutura, os likes. Também observamos que cada comentário das respostas tem uma estrutura bastante semelhante a um comentário normal, sendo apenas distinguível pelas tags do html, o que permitirá que eles sejam filtrados como comentários normais apenas com ligeiras alterações.

Foi possível observar também que sempre que era iniciado uma um comentário e sempre que era iniciada uma resposta dentro de um comentário, estes obedeciam a uma estrutura identificada nas figuras seguintes:

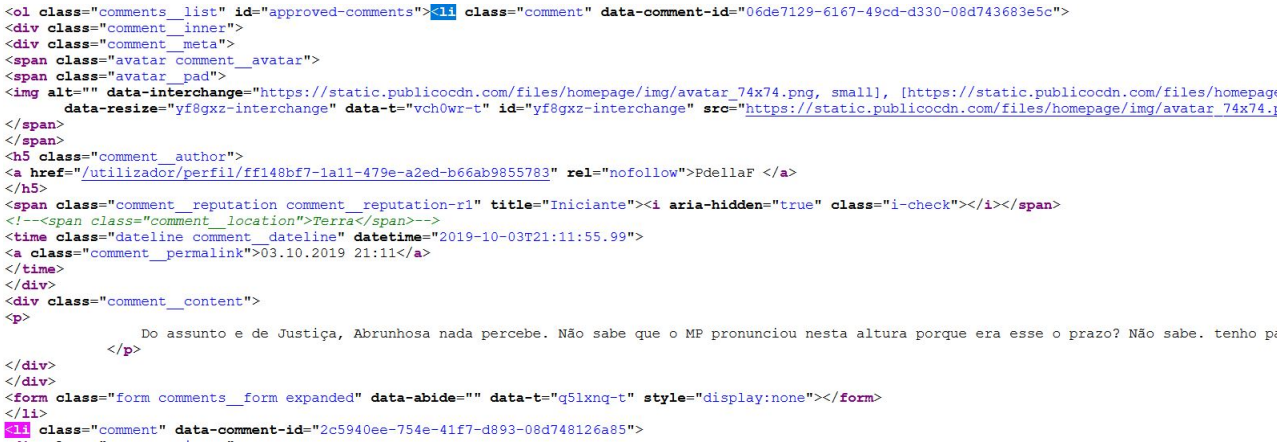


Figura 2.1: Estrutura de um Comentario Singular

Figura 2.2: Estrutura de uma Lista de Respostas dentro de um comentário.

Nestes comentários acima conseguimos identificar bastante bem as tags referentes ao início e fim de um comentário singular (`<li e </li>`), bem como as **tags** para o **nome** (`<h5 class="comment__author"`), **data**(`<time` datetime=") e **conteúdo do comentário** (`<div class="comment__content>`).

Podemos identificar uma **lista de respostas** quando o comentário anterior ao invés de fechar comentário com `</li>`, abre uma `<ol "comments__list">`.

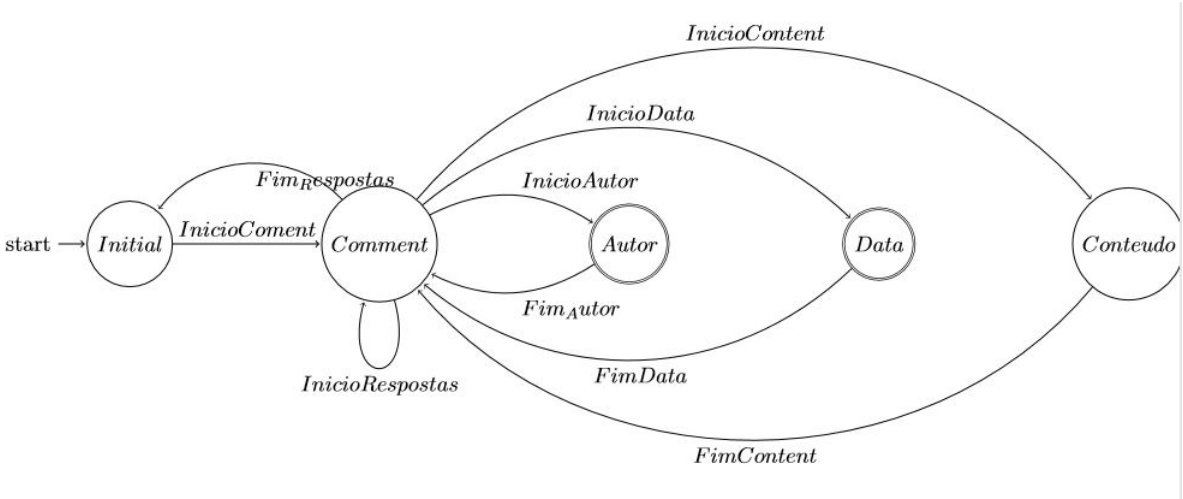
Por fim, como conclusão desta análise inicial, conseguimos ficar bastante esclarecidos da abordagem que iríamos fazer para filtrar a informação necessária. Esta abordagem seria filtrar o nome, data e conteúdo sempre que encontrássemos as respetivas tags, e caso o comentário tivesse a tag das respostas seria aplicado o filtro novamente às respostas até acabar a lista de comentários.

Para facilitar a distinção entre comentário base e respostas, decidimos enviar para a estrutura todas as respostas com o mesmo ID do comentário "pai", e assim por ordem de chegada podemos distinguí-los.

Capítulo 3

Conceção/desenho da Resolução

3.1 Algoritmo



3.1.1 Definições

- Espacos - [ ]*
- InicioContent - `<p>[^A-Za-z0-9.?!"]*`
- InicioData - `<time\ class=\"dateline\ comment__dateline\"\ datetime=\"`
- InicioAutor `rel=\"(nofollow|follow)\">`
- FimComment `<\/li>`

- FimRespostas `<\/ol>`
- FimAutor `<\/a>`
- FimData `\">`
- FimContent `</p>`

Definição do Autómato

Estados do Autómato:

- Initial
- Comment
- Autor
- Data
- Content

Estado Inicial:

- Initial

Funções de Transição:

- **Initial**→ $\{InicioComent\}$ → **Comment**
- **Comment**→ $\{InicioData\}$ → **Data**
- **Comment**→ $\{InicioContent\}$ → **Content**
- **Comment**→ $\{InicioAutor\}$ → **Autor**
- **Autor**→ $\{FimAutor\}$ → **Comment**
- **Data**→ $\{FimData\}$ → **Comment**
- **Content**→ $\{FimContent\}$ → **Comment**
- **Comment**→ $\{FimRespostas\}$ → **Initial**
- **Comment**→ $\{FimComentario\}$ → **Initial**

Capítulo 4

Codificação e Testes

Para a presente secção de codificação e testes baseamo-nos nos capítulos anteriores, pois ambos são extremamente importantes para uma boa implementação desta fase. O capítulo de análise teve a sua relevância pois permitiu-nos ter já uma ideia de que expressões regulares utilizar para filtrar o que desejávamos e o capítulo da concepção da resolução foi relevante pois definimos os autómatos necessários e os contextos precisos para coordenar o processo de filtragem para cada requisito.

4.1 Estrutura de Dados

4.1.1 Lista de Comentários

Quando começamos a desenvolver os primeiros filtros para testar com o ficheiro de input, deparamo-nos com a situação de existirem vários Comentários com varias respostas(Comentário) dentro desse. Seria portanto ideal que armazenássemos todos os comentários e respectivas respostas referentes a um Comentário numa Tabela de Hash.

Adoptando esta abordagem temos que filtrar tudo o que é necessário e só no fim despejar para o ecrã, de forma organizada, tudo aquilo que armazenamos na estrutura.

Desta forma criamos uma estrutura **TodosComentarios**, que contém uma **GHashTable** cuja chave é o id do comentário e o valor é uma outra estrutura **Comentário** que contém o id, o nome do user que fez o comentário, a data, o numero de comentários, o texto do comentário e uma lista de respostas a esse comentário. Estas estruturas são apresentadas nas seguintes imagens.

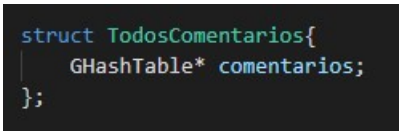


Figura 4.1: Estrutura TodosComentarios.

Figura 4.2: Estrutura Comentário.

4.2 Alternativas, Decisões e Problemas de Implementação

4.2.1 Lista de Comentários

Tendo em conta que um comentário poderia conter ou não e o grupo ter concordado que os comentários deviam estar ordenados por id do comentário original e não pelas respostas a esse comentário, chegamos ao consenso que o ideal seria todas as respostas teriam o mesmo id de um comentário já inserido(anteriormente).

Com isto foi possível, no filtro ter apenas um estado para um comentário e todas as suas respostas, eliminando assim a necessidade de criar um estado para cada resposta. Sendo na inserção da estrutura que é verificada a existência/inexistência do id do comentário presente sabendo assim se se trata de um comentário ou uma resposta se o id já existir ou não.

4.3 Testes realizados e Resultados

4.3.1 Versão output Texto

Aplicando o filtro criado para o ficheiro de input disponibilizado, podem-se verificar resultados com um formato muito parecido ao da seguinte imagem.

Foram imprimidos os comentários, todas as suas respostas correspondentes e o numero de respostas no final de cada comentário.

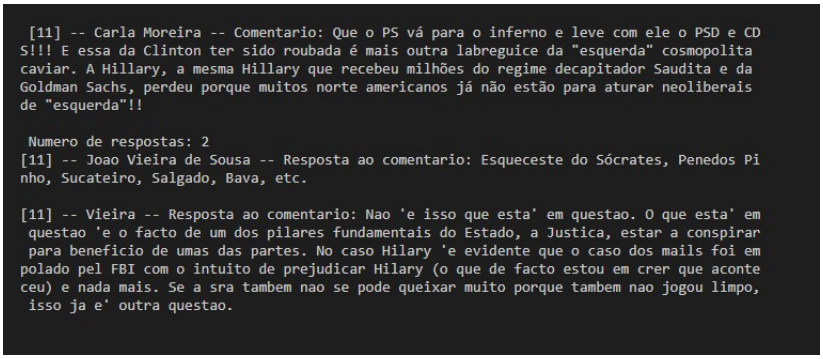


Figura 4.3: Comentário com Id=11 no formato output de texto

4.3.2 Versão output Formato JSON

Aplicando o filtro criado para o ficheiro de input disponibilizado, e com a opção para extrair para um ficheiro JSON é possível verificar resultados com um formato muito parecido ao da seguinte imagem.

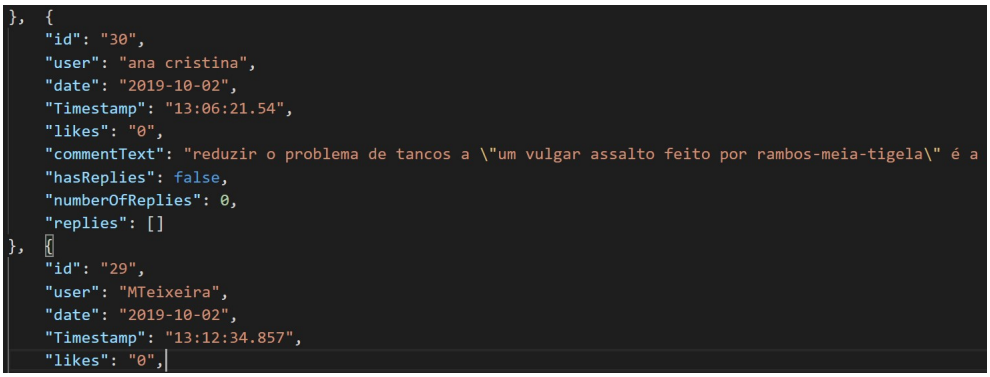


Figura 4.4: Comentário com Id=30 no formato output de ficheiro JSON

Capítulo 5

Conclusão

Tendo em conta os requisitos deste projecto, e o trabalho realizado pelo grupo, achamos que os objectivos fundamentais foram atingidos, sendo estes a capacidade de criar padrões com uso de Expressões Regulares, o entendimento da utilização da ferramenta flex para a criação destes padrões, a capacidade de analisar ficheiros de entrada e criar algoritmos de resolução recorrendo a autómatos.

Ao longo da realização deste projecto o grupo encontrou várias dificuldades, estando estas relacionadas maioritariamente com a melhor forma de estruturar a informação recebida pelo ficheiro e o modo como os filtros iriam funcionar. Entendemos portanto, que esta foi a maior dificuldade pois foi necessário a agilização dos membros do grupo para encontrarem soluções para a filtragem de certos padrões em relação á estrutura final que pretendíamos.

Em jeito de conclusão o grupo acha que todos os requisitos do enunciado foram efectuados com sucesso, e os objectivos principais deste projecto foram atingidos.