Processamento de Linguagens (3º ano de MIEI)

Trabalho Prático 1

Relatório de Desenvolvimento

Catarina Machado (a81047)

Gonçalo Faria (a86264)

João Vilaça (a82339)

31 de Março de 2019

Resumo

O primeiro trabalho prático no âmbito da unidade curricular Processamento de Linguagens consistiu na elaboração de um projeto usando Flex que converte um ficheiro de texto não formatado em vários ficheiros HTML.

Deste modo, no presente relatório, explicamos a forma como desenvolvemos os filtros Flex que nos permitiram recolher os dados do ficheiro não formatado com várias notícias e gerar páginas HTML para cada uma delas, fazer uma listagem de todas as tags que apareciam nas notícias e respetivo número de ocorrências e, por fim, para cada uma das tags ser listada as notícias que a contém, imprimindo também o resultado para páginas HTML.

Os principais objetivos deste trabalho prático foram aumentar a experiência de uso do ambiente Linux e de algumas ferramentas de apoio à programação e aumentar a capacidade de escrever *Expressões Regulares (ER)* para descrição de *padrões de frases*. Para além disso, teve também como objetivo desenvolver, a partir de ERs, sistemática e automaticamente *Processadores de Linguagens Regulares*, que filtrem ou transformem textos com base no conceito de regras de produção *Condiçao-Ação* e utilizar o Flex para gerar *filtros de texto em C*.

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

Área: Processamento de Linguagens

1.1 Enquadramento e Contexto

O **Jornal Angolano - Folha 8, v2** contém milhares de notícias. Cada uma dessas notícias contém tags, id, categorias, título, data, texto e etiquetas.

Assim, o projeto consiste em extrair de um ficheiro *input* com um Jornal Angolano toda essa informação e organizá-la e apresentá-la em páginas html de forma a que possa ser facilmente consultada por qualquer pessoa.

No presente relatório é explicada a forma como cumprimos esse requisito e como geramos as respetivas páginas html com toda a informação, notícias e tags, de forma clara, percetível e apelativa.

1.2 Problema e Objetivo

Numa primeira fase, o problema passa por limpar e normalizar todos esses artigos contidos no Jornal. Em seguida, é necessário criar uma lista das tags encontradas nos artigos, com o respetivo número de ocorrências de cada tag.

Após construída essa lista, que deverá estar presente num ficheiro HTML, o objetivo é colocar um link em cada uma dessas tags identificadas de modo a que, clicando no link, seja apresentado um novo ficheiro HTML contendo o título de cada uma das notícias que contêm a respetiva tag.

Consequentemente, cada um desses títulos deverá também conter uma hiperligação, que reencaminhará o utilizador para outro ficheiro HTML que apresentará a notícia em questão.

1.3 Decisões tomadas

Uma vez que o Jornal, o ficheiro de texto a analisar, é muito extenso, decidimos recorrer à biblioteca multiplataforma GLib de modo a construirmos eficientemente as estruturas de dados com os tipos de dados apropriados e para, consequentemente, podermos utilizar as funções que nos permitem manipular corretamente esses tipos de dados.

Nas estruturas de dados que construímos podem ser guardadas todas as tags e todas as notícias presentes no Jornal Angolano.

1.4 Estrutura do Relatório

O presente relatório está dividido em 5 diferentes capítulos.

No capítulo 1, Introdução, é feito um enquadramento e contextualização do trabalho prático e, em seguida, é feita uma descrição do problema. Por fim, é evidenciado um pequeno ponto no que diz respeito às decisões tomadas.

No capítulo 2, Análise e Especificação, é feita uma descrição informal do problema e uma especificação dos requisitos necessários para uma correta resolução do problema, relevando quais serão as duas grandes tarefas do projeto.

Em seguida, no capítulo 3, Concepção e desenho da Resolução são expostas as estruturas de dados utilizadas e é feita uma descrição detalhada de todo o desenvolvimento do projeto até se obter a solução final. Para além disso, são apresentados máquinas de estados que representam as condições de contexto dos nossos filtros e as expressões regulares que desencadeiam as transições entre esses estados.

Posteriormente, no capítulo 4, Codificação e Testes, são apresentados alguns testes realizados e os resultados obtidos.

Por fim, no capítulo 5, Conclusão, termina-se o relatório com uma síntese e análise do que trabalho realizado.

Capítulo 2

Análise e Especificação

2.1 Descrição informal do problema

Dado um ficheiro de texto bastante extenso com várias notícias, é necessário conseguir extrair de cada notícia as suas tags e construir páginas para cada uma delas, indicando os títulos das notícias que as contêm. Também é necessário extrair as próprias notícias de forma normalizada e a lista de todas as tags encontradas. Toda esta informação tem que ser retirada do extenso ficheiro e dividida por várias páginas html.

2.2 Especificação do Requisitos

De forma a responder devidamente ao problema proposto decidimos dividir o nosso projeto em duas diferentes etapas.

2.2.1 Transformer

A etapa do ***transformer***, que escreve no descritor de saída por defeito todas as publicações necessárias, representadas como um nó xml com nome "pub". Mais precisamente, é feita uma publicação para cada uma das notícias (dividamente formatada), uma publicação para cada uma das tags (cada um desses ficheiros deve ter os títulos das notícias que contêm essa tag e respetivos links para as mesmas) e, por fim, uma publicação com a lista de todas as tags existentes (também com os respetivos links).

2.2.2 Slicer

E no ***slicer***, que separa convenientemente o conteúdo formatado, isto é, cria vários ficheiros, cada um correspondente a cada uma das publicações presentes no texto lido no descritor de entrada por defeito. Essa publicação tanto poderá ser uma notícia, o ficheiro de cada uma das tag ou o ficheiro com a lista de todas as tags.

Capítulo 3

Concepção/desenho da Resolução

3.1 Estruturas de Dados

Para conseguirmos armazenar temporariamente a informação de uma notícia sentimos a necessidade de construir uma estrutura de dados que nos permitisse guardar em diferentes variáveis o id da notícia, o autor (data), as tags, o título, a categoria e as diferentes linhas do texto presente na notícia.

Para isso, construímos a seguinte estrutura de dados, Angola, que tem todas as variáveis necessárias para guardar a informação de uma notícia.

```
typedef struct angola {
    GString * id;
    GString * author;
    GList * tags;
    GString * title;
    GString * category;
    GList * lines;
} *Angola;
```

Tanto o id, como o autor, o título e a categoria são strings, ou seja, simples conteúdo textual.

No caso das tags, uma vez que existe a necessidade de mais tarde termos que delimitar cada uma delas com **<tag>** e **</tag>**, o tipo que as armazena é uma *GList*, para que possam estar divididas e esses dados posteriormente serem mais facilmente tratados.

No caso do texto, decidimos armazena-lo por linhas, ou seja, o texto é também uma *GList* (de linhas), uma vez que quando estivermos a adicionar referências HTML queremos que cada uma dessas linhas seja um parágrafo (isto é, estarem delimitadas por ** ** e ** **).

Para além da necessidade de uma estrutura que permitisse que as diferentes informações de uma notícia estivessem bem organizadas em diferentes variáveis, surgiu também a necessidade de armazenar todas as tags que iam sendo encontradas e os respetivos artigos que as contêm.

```
GHashTable * taghtable;

typedef struct tuple {
    GString * id;
    GString * title;
} *Tuple;
```

Deste modo, foi construída uma variável global ao módulo do tipo *GHashTable* intitulada **taghtable**, com o nome da **tag** na *key* e como *value* o **tuple** apresentado em seguida, que contém o id e o título do artigo.

A decisão de uma *GHashTable* teve origem no facto de existirem notícias repetidas no ficheiro. Como a hash table não permite repetições, estas notícias são ignoradas e o resultado final não será redundante.

3.2 Implementação

3.2.1 Transformer

Tal como mencionado anteriormente, a diretoria ***Transformer*** é a primeira responsável pelo correto *parse* do Jornal Angolano.

Deste modo, foi criado um filtro de texto Flex, chamado *transformer.l*, que com base nas regras de produção Condição-Ação com expressões regulares vai construindo o resultado pretendido.

Impressão da Notícia

Analisando o input disponibilizado (*folha8.OUT.txt*), sabemos que uma notícia começa por um **<pub>** e que possui os campos **#TAG**, **#ID**, seguido pelo campo **categoria**, e, porteriormente, o **título**. Em seguida, temos o campo **#DATE** e, por fim, o **texto** do artigo.

Desta forma, conseguimos definir as condições de contexto necessárias: TAG, IDT, CATEGORY, TITLE, UDATE e TEXT, que correspondem a cada um dos elementos referidos anteriormente, respetivamente.

Deste modo, após termos uma ideia mais concreta da forma como vamos construir as nossas condições, o passo seguinte consistiu na decisão da forma como iriamos representar e guardar cada uma dessas informações para posteriormente conseguirmos extrair a informação do ficheiro da forma desejada. Para tal, recorremos à estrutura de dados Angola, que foi anteriormente exposta e que se encontra no ficheiro *Angola.c*. Esta estrutura de dados permite-nos guardar os dados de cada notícia para depois os podermos imprimir corretamente.

Posto isto, temos todas as ferramentas necessárias para começar a construir o nosso filtro Flex. Tal como constatado ao analisar o ficheiro *input*, sabemos que uma notícia começa por um **<pub>**. Logo, no nosso filtro começamos por colocar essa condição e, quando aparece essa expressão é construída uma instância de "angola".

```
\<pub\>             { a = mkAngola(); }
```

Consequentemente, ao longo da construção do filtro, fomos adicionando ações que visam preencher corretamente a estrutura de dados angola (variável **a**). Iremos exemplicar apenas dois deles uma vez que o funcionamente e o raciocínio por detrás de cada um deles é semelhante.

No caso do *title*, por exemplo, quando nos encontramos nesse estado vamos acumulando no *tbuffer* o título presente no *input*, e quando aparecer o carácter de mudança de linha, é adicionado à estrutura de dados o título identificado (presente em *tbuffer*) e o estado é alterado para INITIAL, de modo a que o filtro volte a poder dar *match* com qualquer uma das condições de contexto existentes. A regra Condição-Ação exposta é a seguinte:

```
<TITLE>.            { g_string_append_c(tbuffer,yytext[0]); }
<TITLE>\n           { tmp = g_string_free(tbuffer,FALSE); addTitle(a,tmp);
                      g_free(tmp); BEGIN INITIAL; }
```

Por exemplo, no caso da data, sempre que aparece "#DATE:" o programa passa a estar no estado UDATE, o estado do correto "parse" da data. Assim, sempre que entramos nesse estado é inicializado um tbuffer, que vai acumulando a data presente no ficheiro *input*. O texto que aparece entre parêntesis retos é considerado lixo e, por tanto, não há qualquer utilidade em armazená-lo. Caso apareça o carácter de mudança de linha significa que a informação da data já terminou e, portanto, pode ser adicionada à nossa estrutura de dados. A variável tbuffer previamente alocada é agora libertada e o estado do programa passa a estar na nossa condição de contexto TEXT.

```
\#DATE\:            {BEGIN UDATE;tbuffer = g_string_new("");}
<UDATE>" "\[.*\]" " {}
<UDATE>.            {g_string_append_c(tbuffer,yytext[0]);}
<UDATE>\n           {tmp = g_string_free(tbuffer,FALSE); addAuthor(a,tmp); g_free(tmp);
                    BEGIN TEXT; tbuffer = g_string_new("");}
```

Por fim, quando a referência pub é fechada (**</pub>**), a notícia é dividamente impressa através da função ***printHTML*** e a estrutura previamente alocada é agora libertada, através da função ***unmkAngola***.

```
\<\/pub\>           {printHTML(a); unmkAngola(a);}
```

Para representar mais objetivamente as condições de contexto criadas (estados) e as ações que originam a transição destas construímos um autómato, Figura 3.1.

Através deste autómato, consegue-se perceber quais são as expressões que originam as transações entre as condições de contexto presentes no nosso filtro *transformer.l*.

A função ***printHTML*** que se encontra na ação de quando o "pub" é fechado, está presente no ficheiro *Angola.c*. Esta função é a responsável por imprimir cada uma das notícias e, como tal, recebe como argumento a estrutura angola (**a**). Todas estas notícias são neste momento impressas num único ficheiro de texto.

Assim, agora que temos todas as informações necessárias prontas a serem utilizadas basta imprimirmos as mesmas (*printf*) com as referências desejadas.

De modo a obtermos um resultado final visualmente apelativo, adicionamos também algum código HTML para além dos requisitos obrigatórios. Por exemplo, no caso da data, para além do **<author_date>** e **</author_date>** adicionamos também o *heading* **h3** e a cor #85C1E9, tal como se pode ver em seguida.

```
printf("\t<author_date><h3>
            \n\t\t<font color='#85C1E9'>%s</font>
          \n\t</h3></author_date>\n", author);
```

Os " \n " e " \t " servem para que a *page source* seja clara e percetível.

É de salientar que dentro de uma notícia cada uma das tags contém também uma hiperligação para a sua página.

Decidimos também apresentar o texto em duas colunas, recorrendo para isso às referências HTML apresentadas em seguida.

```
printf("\t<div style='float: left; width: 50%%;'>\n");
printf("\t<div style='padding: 20px'>\n");
```

Quando tivermos percorrido metade do número total de parágrafos é novamente utilizada a referência HTML mencionada, de modo a que o texto fique então em duas diferentes colunas.

Impressão da Lista de Tags

Tal como já mencionado, houve a necessidade da criação de uma estrutura que armazenasse as tags que iam sendo encontradas ao longo do *parse* do Jornal.

Assim, essa estrutura vai sendo preenchida quando o artigo está a ser impresso, ou seja, quando o programa está a correr a função ***printHTML***, mencionada na secção anterior, Secção 3.2.1. Esta função, quando está a imprimir uma notícia, mais precisamente na altura em que está a imprimir as tags da mesma, recorre à função auxiliar ***addlink*** para cada uma das tags presentes. A função ***addlink*** recebe como parâmetro a tag em questão, o id e o título da notícia em curso.

Deste forma, depois de estarem impressas todas as notícias presentes no Jornal, temos a nossa *GHash-Table* preenchida e pronta a ser utilizada.

De modo a cumprirmos os dois requisitos em falta, (imprimir a lista de todas as tags e imprimir uma lista das notícias que contêm cada uma das diferentes tags) recorremos a duas macros, apresentadas em seguida, e que estão presentes no ficheiro *Angola.h*.

```
#define START_TAGGING tag()
#define STOP_TAGGING trace()
```

Desta forma, na nossa função **main** estas macros são utilizadas tal como se pode ver em seguida. Com o START_TAGGING criamos e inicializamos a *GHashTable* e o STOP_TAGGING é o responsável por tratar dos dois requisitos em falta.

```
int main(){
    START_TAGGING;
    yylex();
    STOP_TAGGING;
    return 0;
}
```

Assim, o START_TAGGING é o responsável por invocar a função ***trace***, que imprime as páginas html em falta e que liberta a memória da *GHashTable* no final.

```
void trace(){
    printTagsHTML();

    printTagTitlesHTML();

    g_hash_table_destroy(taghtable);
}
```

Para o caso de imprimir a lista das tags a função utilizada é a ***printTagsHTML***.

Esta função começa por imprimir o título e o cabeçalho pretendido para a página HTML Tags.html e, em seguida, vai percorrendo as *keys* da *GHashTable* e imprimindo em 3 colunas as tags encontradas. Para cada uma das tags é impresso o nome da tag (mesmo que contenha espaços), sendo que esse nome contém uma hiperligação para uma página com o nome da tag (mas, neste caso, com os espaços substituídos por hífens, devido a limitações de endereçamento).

É também impresso o número de ocorrências de cada tag, que corresponde ao tamanho da lista do *value* de cada uma das *keys*.

Depois de toda a função ***printTagsHTML*** ser processada, no ficheiro já existente com todas as notícias é adicionado no final a "pub" **tags**, com a lista de todas as tags, as respetivas hiperligações e número de ocorrências.

Impressão da Tag com lista de notícias

Em consequência do que foi explicado na secção anterior, Secção 3.2.1, com recurso à macro START_TAGGING, que chama a função ***trace()*** (já apresentada), depois dessa função invocar a função ***printTagsHTML***, responsável por imprimir a lista das tags, invoca a função ***printTagTitlesHTML*** que colmatará o último requisito proposto, imprimir para cada uma das tags a lista dos títulos dos artigos que a contêm.

Esta função segue o mesmo raciocínio das funções ***print*** já mencionadas. Com recurso à *GHashTable*, percorre cada uma das tags (**key**) e para cada uma delas cria um novo "pub" com o id da respetiva tag (substituindo os espaços por hífens), e no conteúdo imprime a lista das notícias (*title*) com uma hiperligação para a mesma (obtido através do id da notícia) (**value**).

Tal como é possível verificar nas Figuras 4.1, 4.2 e 4.3, e também nos resultados textuais expostos no presente relatório, no final de cada uma das páginas é apresentada uma hiperligação "Regressar para Tags.html" que permite regressar para a página inicial com a lista de todas as tags.

3.2.2 Slicer

Depois do ficheiro exemplo de input *folha8.OUT.txt* passar pelo filtro *transformer.l* temos um ficheiro com todas as "pub" pedidas. Assim, resta-nos agora dividir todas elas, criando um ficheiro html para cada uma. Na diretoria ***Slicer*** encontra-se o ficheiro *glue.l*, responsável por resolver o problema através de expressões regulares e de regras de Condição-Ação.

Na Figura 3.2 encontra-se um autómato, onde se encontram representados as condições de contexto como sendo estados e as ações como a ação que despulta o ínicio desse estado.

O trabalho efetuado por este filtro é bastante simples. Assim que encontra a condição "pub" o filtro entra no estado NAME. Nesse estado, é criado um ficheiro com o nome pretendido (**id**) e inicializado com as tags standard de inicialização de um ficheiro HTML (<!DOCTYPE html> e <html>).

Em seguida, o filtro vai para o estado COPY, que vai copiando todo o conteúdo para o ficheiro HTML anteriormente criado. Quando o "pub" é fechado, o ficheiro html é também fechado e o filtro volta para o estado inicial.

```
"<pub id=\""                    {buffer[0]=dir[0]='\0';
                                 BEGIN NAME;}
<NAME>\">                       {strcat(buffer,".html");
                                 sprintf(dir,"%s%s",dirout,buffer);
                                 fdes = fopen(dir,"w");
                                 fprintf(fdes,"<!DOCTYPE html>\n<html>\n");
                                 BEGIN COPY;}
<NAME>.                         {strcat(buffer,yytext);}
<COPY>"</pub>"                  {fprintf(fdes,"\n</html>");
                                 fclose(fdes);
                                 BEGIN INITIAL;}
<COPY>.|\n                      {putc(yytext[0] ,fdes);}
<*>.|\n                         {}
```

3.2.3 Makefile

Para uma compilação fácil e simples do programa criamos uma Makefile. Desta forma, tanto na diretoria ***Transforme*** como na diretoria ***Slice***, foi criada uma Makefile que visa compilar o filtro *Flex* e, em seguida, compilar o código *C* resultado do *Flex*. No caso da diretoria ***Transformer***, é também necessário utilizar o package da biblioteca GLib e os includes por nós criados.

Na diretoria principal do trabalho, que contém as diretorias anteriormente mencionadas, temos uma terceira Makefile. Essa Makefile é responsável por correr as outras duas Makefiles, gerando os executáveis **transformer** e **slicer**, e por finalmente gerar o executável final, intitulado **go**.

Assim, para correr o nosso programa basta escrever **make** no terminal, na diretoria do trabalho e executar o executável gerado (**go**), com o ficheiro *input* desejado.

```
./go < folha8.OUT.txt
```

Por definição, os ficheiros html resultantes da execução do programa vão para uma nova pasta chamada **out**. No entanto, existe a opção de mudar o nome da pasta. Para tal, basta que ao fazer "make" se indica o opção DIROUT pretendida, tal como se pode ver em seguida.

```
make DIROUT=nomepretendido
```

A Makefile também possui a opção de *clean*, que limpa o projeto removendo os binários e a pasta com o *output*.

3.3 Outros extras

Para além do embelezamento do output e das Makefiles criadas, decidimos ainda responder à questão 3) de outra forma.

Adicionalmente, incluímos na entrega final na diretoria ***TagsCounter***, o filtro que criamos para realizar uma exploração inicial dos dados. Este filtro contabiliza o número de ocorrências das tags, no entanto não considera a possibilidade de notícias repetidas nos dados. Esta contagem foi feita através da criação de um histograma com a estrutura de dados *MultiSet* que nós criamos.

Para a compilação deste programa foi também criada uma Makefile.

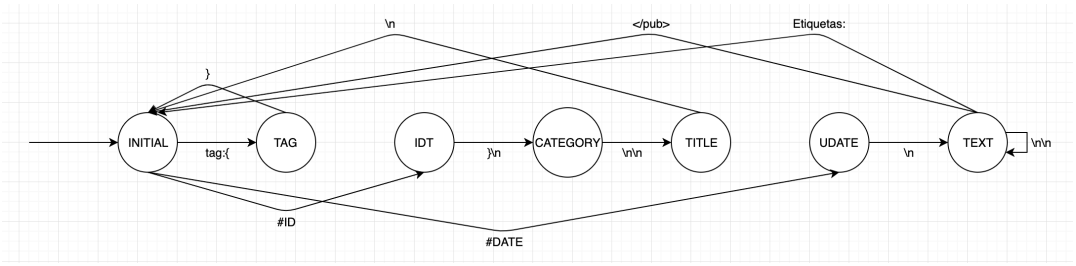

Figura 3.1: Autómato Transformer

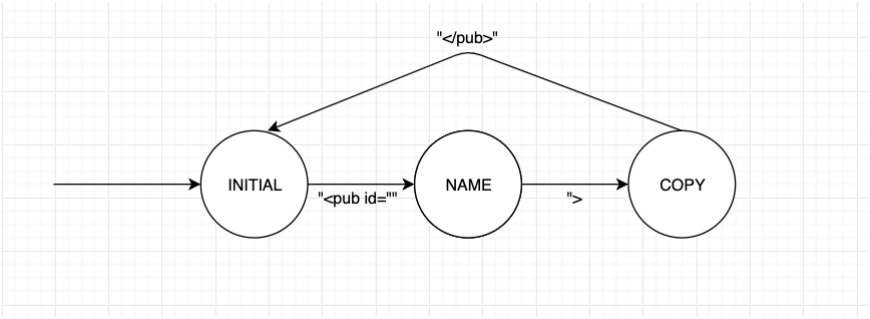

Figura 3.2: Autómato Slicer

Capítulo 4

Codificação e Testes

4.1 Testes realizados e Resultados

4.1.1 Impressão da Notícia

Dada a seguinte notícia:

```
    <!-- ====================2014/100-milhoes-de-dolares-em-droga/index.html -->
<pub>
#TAG: tag:{Apreensão} tag:{Droga} tag:{EUA} tag:{São Tomé e Príncipe} tag:{narcotráfico}
#ID:{post-1643 post type-post status-publish format-standard has-post-thumbnail hentry
category-lusofonia tag-apreensao tag-droga tag-eua tag-sao-tome-e-principe tag-narcotrafico}
Lusofonia

100 milhões de dólares em droga


PARTILHE VIA:
#DATE: [116eb] Redacção F8 | 30 de Setembro de 2014
[aaa15]

Um navio de carga com bandeira são-tomense que transportava mais de cem milhões de dólares
em droga foi apresado há cerca de três semanas pela marinha dos EUA, revelou hoje uma
fonte governamental são-tomense.

Segundo a mesma fonte, o navio de nome Borocho, capturado nas águas da América Central,
continha muita droga, cujo valor ronda mais de 100 milhões de dólares", além de outras
mercadorias ilegais.
A embarcação estava matriculada sob o número 003651 nas águas da América Central.

Num comunicado do Conselho de Ministro distribuído esta semana indica-se que \o  Governo
foi informado que no dia 07 deste mês as competentes autoridades da Guarda Costeira em
parceria com as cooperações de países amigos deram início a diligências para interceptar
no navio de carga de nome Borocho, supostamente registado com bandeira são-tomense".

O governo não dá mais pormenores, adiantando apenas que o referido navio é desconhecido do
```

Instituto Marítimo e Portuário de São Tomé e Príncipe (IMAP-STP), a única instituição vocacionada para registar navios e atribuir bandeira são-tomense.

"Neste momento, o referido navio com registo ilegal foi apresado e encontra-se sob custódia com estatuto de embarcação sem nacionalidade", refere-se no comunicado do governo são-tomense.

Sublinha-se ainda que \perante a gravidade deste caso", que afecta \a imagem do país, o conselho de ministros insta as autoridades competentes do sector no sentido de prosseguirem com as investigações, identificar cumplicidades e prevaricadores para efeitos de procedimento disciplinar e criminal".

A nível interno em São Tomé, as autoridades não revelam pormenores deste caso, mas sabe-se que o navio transportava 700 quilos de cocaína e que foram detidos 13 tripulantes.

A embarcação é de origem do Panamá e a documentação que as autoridades do arquipélago dizem ser falsas foram emitidos em 23 de Julho último e é válida até a mesma data de 2018.

z

Etiquetas: ApreensãoDrogaEUASão Tomé e Príncipenarcotráfico
</pub>

O *output* textual obtido é o seguinte:

```
<pub id="post-1643">
<head>
        <title>100 milhões de dólares em droga</title>
</head>
<body>
        <h1><p align='center'>
                <font color='#2874A6'>100 milhões de dólares em droga</font>
        </p></h1>
        <div style='padding: 20px'>
        <author_date><h3>
                <font color='#85C1E9'>
                Redacção F8 | 30 de Setembro de 2014
                </font>
        </h3></author_date>
        <p><tags><b>Tags:</b>
                <a href="Apreensão.html"><tag>#Apreensão</tag></a>
                <a href="Droga.html"><tag>#Droga</tag></a>
                <a href="EUA.html"><tag>#EUA</tag></a>
                <a href="São-Tomé-e-Príncipe.html">
                    <tag>#São Tomé e Príncipe</tag></a>
                <a href="narcotráfico.html"><tag>#narcotráfico</tag></a>
        </tags></p>
        <p><category><b>Categoria:</b> Lusofonia</category></p>
        </div>
        <text>
        <div style='float: left; width: 50%;'>
        <div style='padding: 20px'>
                <p align='justify'>Um navio de carga com bandeira...</p>
                <p align='justify'>Segundo a mesma fonte, o navio de nome...</p>
                <p align='justify'>Num comunicado do Conselho de Ministro...</p>
                <p align='justify'>O governo não dá mais pormenores, ...</p>
                <p align='justify'>\Neste momento, o referido navio com...</p>
        <br><br>
    <a href="tags.html"><tag>Regressar para Tags.html</tag></a>
        </div></div>
        <div style='float: left; width: 49%;'>
        <div style='padding: 20px'>
                <p align='justify'>Sublinha-se ainda que \perante a...</p>
                <p align='justify'>A nível interno em São Tomé, as...</p>
                <p align='justify'>A embarcação é de origem do Panamá e a...</p>
        </div></div>
        </text>
</body>
</pub>
```

100 milhões de dólares em droga

Redacção F8 — 30 de Setembro de 2014

Tags: #Apreensão #Droga #EUA #São Tomé e Príncipe #narcotráfico

Categoria: Lusofonia

Um navio de carga com bandeira são-tomense que transportava mais de cem milhões de dólares emdroga foi apresado há cerca de três semanas pela marinha dos EUA, revelou hoje uma fontegovernamental são-tomense.

Segundo a mesma fonte, o navio de nome Borocho, capturado nas águas da América Central, continhamuita droga, cujo valor ronda mais de 100 milhões de dólares", além de outras mercadorias ilegais.A embarcação estava matriculada sob o número 003651 nas águas da América Central.

Num comunicado do Conselho de Ministro distribuído esta semana indica-se que "o Governo foiinformado que no dia 07 deste mês as competentes autoridades da Guarda Costeira em parceria com ascooperações de países amigos deram início a diligências para interceptar no navio de carga de nomeBorocho, supostamente registado com bandeira são-tomense".

O governo não dá mais pormenores, adiantando apenas que o referido navio é desconhecido doInstituto Marítimo e Portuário de São Tomé e Príncipe (IMAP-STP), a única instituição vocacionadapara registar navios e atribuir bandeira são-tomense.

"Neste momento, o referido navio com registo ilegal foi apresado e encontra-se sob custódia comestatuto de embarcação sem nacionalidade", refere-se no comunicado do governo são-tomense.

Sublinha-se ainda que "perante a gravidade deste caso", que afecta "a imagem do país, o conselho deministros insta as autoridades competentes do sector no sentido de prosseguirem com asinvestigações, identificar cumplicidades e prevaricadores para efeitos de procedimento disciplinare criminal".

A nível interno em São Tomé, as autoridades não revelam pormenores deste caso, mas sabe-se que onavio transportava 700 quilos de cocaína e que foram detidos 13 tripulantes.

A embarcação é de origem do Panamá e a documentação que as autoridades do arquipélago dizem serfalsas foram emitidos em 23 de Julho último e é válida até a mesma data de 2018.

Regressar para Tags.html

Figura 4.1: Artigo 100 milhoes de dolares em droga

O *output* html do código acima apresentado é o apresentado na Figura 4.1.

4.1.2 Impressão da Lista de Tags

O resultado em texto obtido correspondente à "pub" de todas as tags existentes é o seguinte.

```
<pub id="tags">
<head>
        <title>Tags</title>
</head>
<body>
        <h1><p align='center'><font color='#2874A6'>Tags</font></p></h1>
        <div style='float: left; width: 33%;'>
        <ul>
                <li><a href="Raul-Araújo.html">Raul Araújo</a>
                    <font color='#2874A6'>(2)</font></li>
                <li><a href="Nobel-da-Paz-2014.html">Nobel da Paz 2014</a>
                    <font color='#2874A6'>(1)</font></li>
                ...
    </ul>
    </div>
    <div style='float: left; width: 33%;'>
    <ul>
                <li><a href="Barack-Obama.html">Barack Obama</a>
                    <font color='#2874A6'>(1)</font></li>
                <li><a href="ficheiro.html">ficheiro</a>
                    <font color='#2874A6'>(1)</font></li>
                ...
        </ul>
    </div>
    <div style='float: right; width: 33%;'>
    <ul>
                <li><a href="SOS-Racismo.html">SOS Racismo</a>
                    <font color='#2874A6'>(1)</font></li>
                <li><a href="União-dos-Escritores-Angolanos.html">
                    União dos Escritores Angolanos
                    </a><font color='#2874A6'>(1)</font></li>
        ...
</body>
</pub>
```

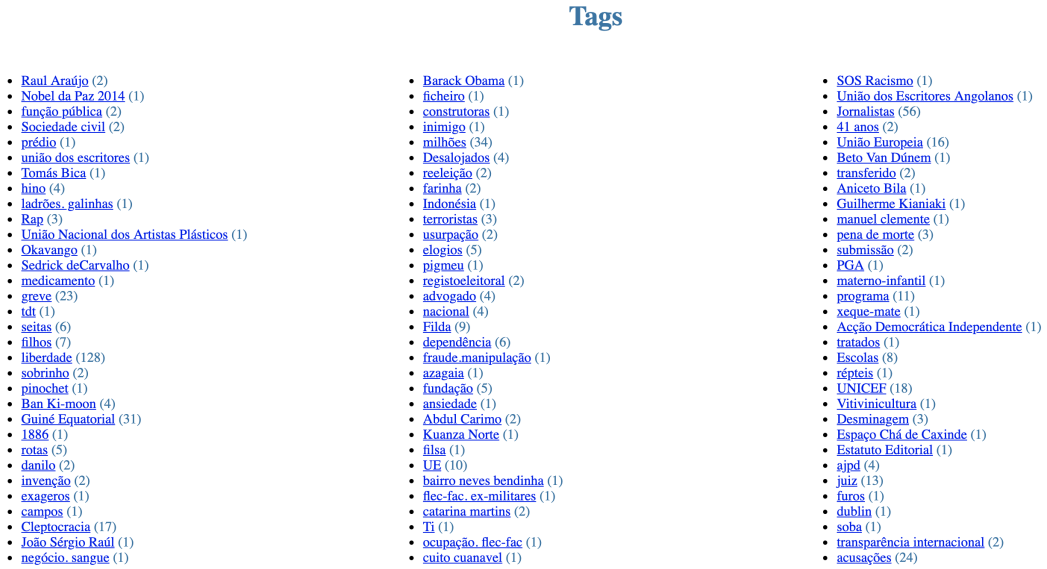

Figura 4.2: Tags.html

O resultado visual resultante é o apresentado na Figura 4.2.

4.1.3 Impressão da Tag com lista de notícias

Para a tag “Raul Araújo” o resultado textual obtido é o apresentado em seguida.

```
<pub id="Raul-Araújo">
<head>
        <title>Raul Araújo</title>
</head>
<body>
        <h1><p align='center'><font color='#2874A6'>Raul Araújo</font></p></h1>
        <h2>Número de ocorrências: <font color='#2874A6'>2</font></h2>
<ul>
        <li><a href="post-1175.html">
        Reorganização judiciária em Portugal? Uma anarquia!
        </a></li>
        <li><a href="post-1283.html">
        Angola com novo Código Penal ainda este ano
        </a></li>
</ul>
        <br><br><br><br><br><br><br>
        <a href="tags.html"><tag>Regressar para Tags.html</tag></a></body>
</pub>
```

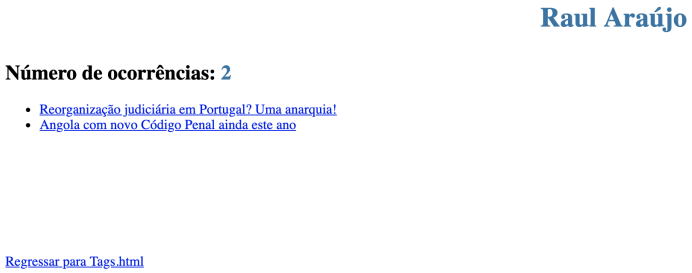

Figura 4.3: Tag Raul Araujo

Visualmente, a página da tag mencionada é a apresentada na Figura 4.3.

Capítulo 5

Conclusão

Através do desenvolvimento deste projeto conseguimos melhorar os nossos conhecimentos relativamente à construção de filtros Flex, de expressões regulares e de regras Condição-Ação. A utilidade destes filtros foi realmente notória, uma vez que conseguimos a partir de um ficheiro com milhares de notícias não formatadas, confuso e extenso, gerar várias páginas html com toda essa informação de forma organizada e apelativa, o que revela que a utilização destes filtros faz realmente a diferença quando o nosso objetivo é apresentar informação, analisar dados, recolher dados, entre muitas outras utilidades.

Abrindo o ficheiro Tags.html (Figura 4.2) no *browser*, é possível visualizar a lista de todas as tags em três diferentes colunas. Tomamos a decisão de as organizar em três colunas pois são imensas tags e desta forma não será necessário fazer tanto *scroll* para as conseguir encontrar a todas. Esta página, na nossa opinião, é bastante apelativa e é bastante claro o objetivo da mesma - apresentar a lista de todas as tags, com a informação do número de ocorrências de cada uma no Jornal Angolano.

Assim, através desta página, é possível clicar em alguma das presentes tags e observar os títulos de todas as notícias que contêm essa tag. Nesta nova página (Figura 4.3), também é permitido regressar à página anterior através da hiperligação presente no canto inferior esquerdo da página. É possível ainda clicar nos títulos das notícias e visualizar o conteúdo da respetiva notícia.

Por fim, dentro da página da tag, o utilizador pode ainda navegar para uma das notícias. A notícia (Figura 4.1) é também apresentada num separador do *browser*, onde o utilizador pode ver o seu título e a data da mesma. Pode também ver as tags da notícia, que contém uma hiperligação para a respetiva página, caso o utilizador pretende navegar para ela. O texto é apresentado num formato de colunas, tal como num jornal em formato de papel, para que seja mais rápido de ler e para que o utilizador possa ter um *overview* de toda a notícia.

Com este extenso ficheiro seria possível realizar muitos outros filtros que permitiram que os dados presentes no Jornal fossem analisados e apresentados de várias outras formas, fornecendo mais funcionalidades ao utilizador.

Concluindo, através do enorme ficheiro de texto, conseguimos cumprir todos os requisitos propostos e construir um agradável fluxo de páginas html de apresentação de notícias e tags, que certamente fez com que o Jornal Angola ainda mais satisfatório de ler.