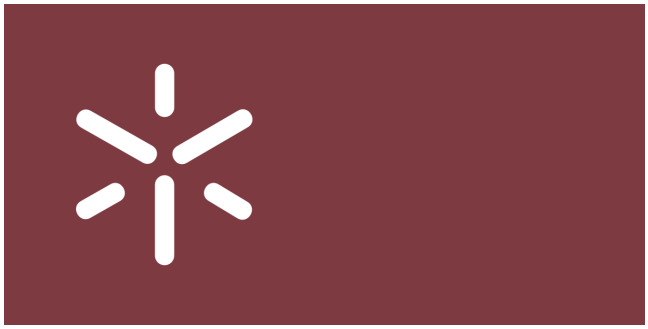


Universidade do Minho

Mestrado em Engenharia Informática

Scripting no Processamento de Linguagem Natural

Website Processing

Autores:

Frederico Pinto	A73639
Rui Vieira	A74658

1 de Julho de 2019

Conteúdo

1 Introdução

Com o objetivo de melhorar os nossos conhecimentos sobre os temas estudados na unidade curricular de *Scripting no Processamento de Linguagem Natural* foi nos proposto uma série de enunciados, optamos então por escolher o enunciado 6, que se baseia na criação de ficheiros *TMX* a partir de técnicas de *web scraping* utilizando o *Beautiful Soup*.

Neste relatório, iremos numa fase inicial falar sobre a estrutura dos ficheiros *TMX* e posteriormente aprofundar os objetivos propostos. Por fim vamos falar sobre a nossa proposta de implementação para cumprir com sucesso esses objetivos.

2 Ficheiros TMX

Translation Memory, *TM*, é uma tecnologia de linguagem que permite a tradução de segmentos (frases, parágrafos ou frases) de documentos, pesquisando segmentos semelhantes numa base de dados e sugerindo correspondências de idiomas diferentes que se encontram presentes nessa base de dados.

É nesse contexto que surgem os ficheiros *TMX*, *(Translation Memory eXchange)*, que são um padrão *XML* aberto e neutro de fornecedor. Têm como objetivo facilitar a troca de dados sobre *translation memory* entre ferramentas ou fornecedores de tradução apresentando pouca ou até nenhuma perda de dados importantes durante esse processo, o que é uma grande vantagem.

2.1 Estrutura

Como foi falado acima, o *TMX* apresenta uma sintaxe parecida ao *XML*, apresentando elementos e atributos únicos, contendo também várias regras para construir um ficheiro válido. Sendo uma quantidade grande de elementos, atributos e regras, colocamos nas referências uns *links* para se poder verificar todos os elementos, atributos e regras que o *TMX* contêm.

Contudo, num sentido bastante abstrato, um ficheiro *TMX* possui um elemento base *<tmx>* que possui dois elementos, um *<header>* que contêm atributos sobre o ficheiro, e um elemento *<body>* que contêm a informação relevante. Esse elemento *<body>* contêm uma lista de elementos *<tu>* que são *translation units*, possuindo cada um, elementos *<props>* e *<note>*, que representam informação sobre essa *translation unit*. Para além disso cada *<tu>* possuí elementos *<tuv>* que são *translation unit variant*, este elemento possuí um filho que é o elemento *<seg>* que contêm o segmento de texto que o *<tuv>* possuí.

Apresentamos de seguida um exemplo de um ficheiro *TMX*, contendo os elementos que falamos acima.

```
<?xml version="1.0" encoding="UTF-8" ?>
<tmx version="1.4">
  <header adminlang="en"
     creationtool="Heartsome Dictionary Editor"
     creationtoolversion="1.0"
     datatype="tbx"
     o-tmf="unknown"
     segtype="block"
     srclang="en"/>
  <body>
    <tu origin="tbx" tuid="1108600011738">
      <tuv xml:lang="en">
        <prop type="administrativeStatus">admittedTerm-admn-sts</
    prop>
        <prop type="termType">entryTerm</prop>
        <prop type="usageNote">Colloquial use term</prop>
        <note>Informal salutation</note>
        <seg>Hello</seg>
      </tuv>
      <tuv xml:lang="es">
        <prop type="administrativeStatus">admittedTerm-admn-sts</
    prop>
        <prop type="termType">entryTerm</prop>
        <prop type="usageNote">Termino de uso coloquial</prop>
        <note>Saludo informal</note>
        <seg>Hola</seg>
      </tuv>
    </tu>
  </body>
</tmx>
```

Listing 1: Exemplo de um ficheiro *TMX*.

3 Objetivos

Após análise da estrutura do ficheiro *TMX*, podemos definir os objetivos do nosso trabalho, ou seja, que informação podemos encontrar em *websites* e colocar no ficheiro *TMX* criado.

Decidimos então, produzir um *script* capaz de traduzir uma ou mais palavras para idiomas escolhidos pelo utilizador, que para além disso tem que ser capaz de criar uma *translation unit* por cada significado semântico que essa palavra possa ter, incluindo a sua significação e palavras sinónimas.

4 Implementação

Como proposto no enunciado para resolver este problema temos que fazer *web scraping* em *Python* utilizando o *Beautiful Soup*. O *Beautiful Soup* é uma biblioteca para o *Python* que retira informação de ficheiros *HTML* e *XML*, utilizando um *parser* que permite obter maneiras de navegar, procurar e modificar a àrvore analisada. Sem esta ferramenta, um *developer* perderia horas analisando toda a informação "à mão".

De acordo com os objetivos propostos na secção anterior e após realizarmos uma pesquisa, concluímos que necessitamos de fazer *scraping* em dois *websites*. Iremos utilizar o *Linguee* para extrair as traduções em todas os idiomas que este fornece e o *Lexico* para verificar a significação das palavras e os seus sinónimos.

De seguida, iremos falar sobre qual foi a nossa estratégia de implementação para cada um deles. Contudo, é de salientar que as palavras que o nosso *script* recebe têm que ser em inglês, pois é a língua mais suportada do *Linguee*, sendo possível traduzir para os seguintes idiomas.

- Alemão
- Francês
- Espanhol
- Chinês
- Russo
- Japonês
- Português
- Italiano
- Holandês
- Polaco
- Sueco
- Dinamarquês
- Finlandês
- Grego
- Checo
- Romeno
- Húngaro
- Eslovaco
- Búlgaro
- Esloveno
- Lituano
- Letão
- Estónio
- Maltês

A procura no *Lexico* é efetuada com a palavra em inglês, pois apresenta um excelente suporte.

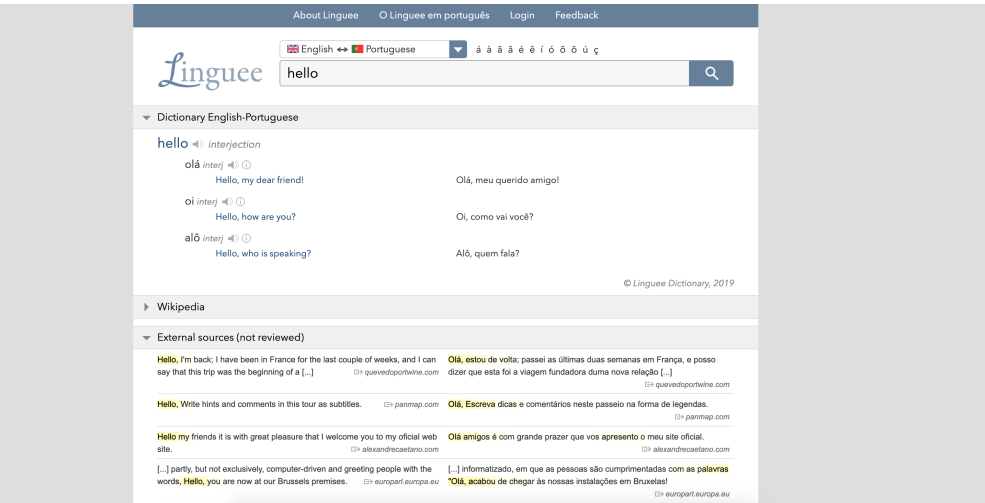
4.1 *linguee.pt*

Em primeiro lugar é importante analisar o *URL* que faz o pedido ao *endpoint* pela informação e quais são os campos a alterar de maneira a efetuarmos nós os pedidos com a informação que desejamos.

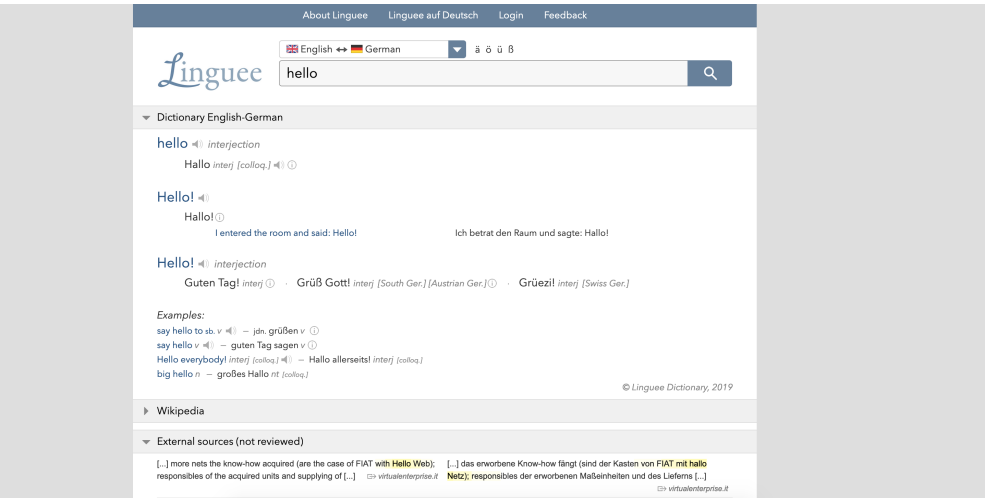
```
https://www.linguee.com/english-portuguese/search?query=hello
```

Listing 2: Exemplo de URL de pedidos ao Linguee.

Esse pedido devolve a seguinte página *HTML*:



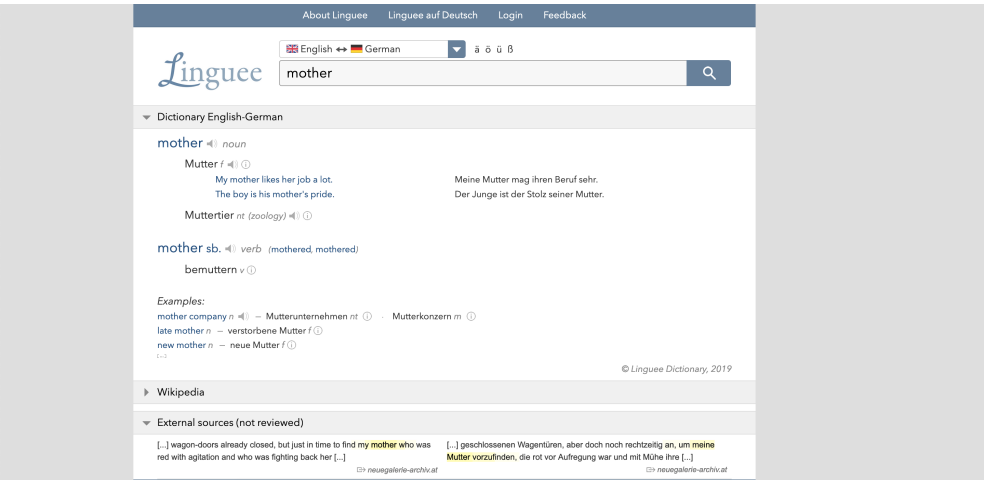
Após analisarmos o *URL* podemos concluir que existem componentes que sendo alterados, são pedidos válidos e contêm a informação necessária para obtermos aquilo que desejamos. O primeiro caso é o idioma para o qual queremos traduzir, o segundo componente do *URL* (*english-portuguese*), se substituirmos por *english-german*, obtemos a seguinte página, que apresenta a tradução de *hello* para alemão.



Para podermos variar o idioma da tradução, guardamos um *array* com todos os idiomas que o *Linguee* suporta e iteramos o mesmo, para fazer variar

o *URL* do pedido *HTTP*.

Por fim, e o mais óbvio, se alterarmos a variável *query* presente no *URL* para qualquer outra palavra em inglês, obtemos a tradução dessa palavra para o idioma indicado no *URL*. Alterando a variável *query* para *mother*, obtemos o seguinte resultado.



Após analisarmos como poderíamos navegar pelo *Linguee* a nível de pedidos *HTTP*, temos agora que analisar como chegar à informação necessária presente na página *HTML* com o *Beautiful Soup*. De seguida mostramos um excerto da página que contêm informação necessária.

```
<html>
<head>
    ...
</head>
<body>
    ...
    <div id="dictionary">
        <h1>...</h1>
        <div class="isMainTerm">
            <div class="exact">
                <div class="lemma">
                    <div>
                        <h2 class="line lemma_desc" lid="EN:mother25752">
                            <span class="tag_lemma">
                                <a class="dictLink" rel="nofollow" href="/english-german/translation/mother.html">mother</a>
                            <span class="tag_wordtype">noun</span>
                            </span>
                        </h2>
                        <div class="lemma_content">
                            <div class="meaninggroup sortablemg" gid="0">
                                <div class="translation_lines">
                                    <div class="translation sortablemg featured">
                                        <h3 class="translation_desc">
                                            <span class="tag_trans" bid="10000250407" lid="DE:Mutter24716"><a id="dictEntry10000250407"
                                            href="/german-english/translation/Mutter.html" class="dictLink featured">Mutter</a>
                                            </span>
                                        </h3>
                                    </div>
                                    <div class="translation sortablemg featured">
                                        <h3 class="translation_desc">
                                            <span class="tag_trans" bid="10000247790" lid="DE:Muttertier60356">
```

```
                                        <a id="dictEntry10000247790" href="/german-
    english/translation/Muttertier.html"
                                        class="dictLink featured">Muttertier</a>
                                        </span>
                                    </h3>
                                </div>
                            </div>
                        </div>
                    </div>
                <div class="lemma">
                    (que contem outras traduçoes para diferentes significados
    semanticos da palavra)
                </div>
            </div>
        </div>
```

Listing 3: Página HTML fornecida pelo Linguee.

Após analisar todo o *HTML* recebido, para irmos buscar a informação importante, precisamos de ter o seguinte *workflow*.

```
1
2   # Fazer o pedido com o URL previamente construido
3   response = requests.get(url).content
4
5   # Analise da pagina html pelo Beautiful Soup
6   soup = BS(response, 'html.parser')
7
8   # Encontrar a DIV que contem o id dictionary
9   dictionary = soup.find('div', id="dictionary")
10
11  # Encontrar dentro da DIV dictionary a DIV com classe exact
12  exact = dictionary.find('div', 'exact')
13
14  # Encontrar dentro da DIV exact todas as DIVS com classe
    lemma
15  lemma = exact.findAll('div', 'lemma')
16
17  # Por cada DIV encontrada acima
18  for lem in lemma:
19
20      # Encontrar o span que contem a classe tag_lemma para ir
    buscar a tag_wordtype
21      word = lem.find('span', 'tag_lemma')
22
23      # Descobrir o span que contem o tipo semantico da palavra
     que esta presente neste DIV lem
24      tag_wordtype = word.find('span', 'tag_wordtype')
25
26      word_type = tag_wordtype.text
27
28      # Ir buscar a primeira tradução que estiver contida na
    DIV lem
29      translation = lem.find('span', 'tag_trans')
```

Listing 4: Implementação com o Beautiful Soup para o *Linguee*.

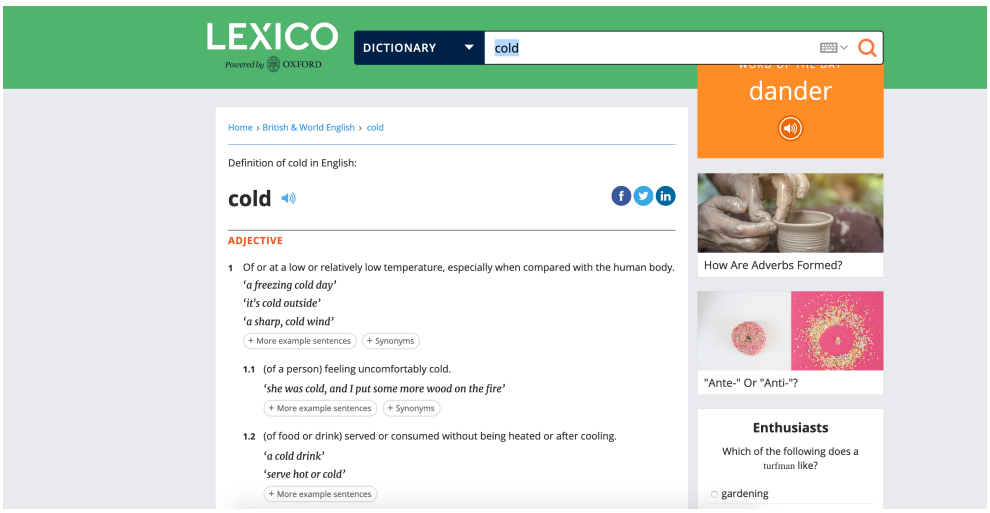
4.2 *lexico.com*

Tal como o *Linguee* numa primeira fase temos que analisar os *URLs* que fazem as conexões aos *endpoints* que queremos atingir. Sendo assim apresentamos de seguida, o *URL* que permite ver a definição de *hello* no *Lexico*.

```
https://www.lexico.com/en/definition/cold
```

Listing 5: Exemplo de URL de pedidos ao Lexico.

Esse pedido devolve a seguinte página:



Como podemos verificar, a única parte variável deste *URL* é a ultima parte, *cold*. Substituindo pela palavra desejada, obtemos a página que contêm a definição dessa palavra.

Sendo assim, mostramos de seguida o *HTML* que compõe a página para sabermos como podemos obter a informação que necessitamos, que é a significação e as palavras sinónimos.

```
<html>
<head>
    ...
</head>
<body>
    ...
    <section class="gramb">
        <h3 class="ps pos"><span class="pos">adjective</span></h3><span class="
    transitivity"></span>
        <ul class="semb">
            <li>
                <div class="trg">
                    <p>
                        <span class="iteration">1</span>
                        <span class="ind">Of or at a low or relatively low temperature,
                        especially when compared with the human body.</span>
                    </p>
                    <div class="exg"> ... </div>
                    <div class="examples"> ... </div>
                    <div class="synonyms">
                        <div class="moreInfo"><button data-behaviour="ga-event-synonyms"
                            data-value="expand/collapse">Synonyms</button></div>
                        <div class="exg">
```

```
                                <div>
                                    <strong class="syn">chilly</strong>
                                    <span class="syn">, cool, freezing, icy, snowy, icy-cold,
                                        glacial, wintry, crisp, frosty, frigid, bitter, bitterly cold, biting, piercing, numbing,
                                        sharp, raw, polar, arctic, Siberian
                                    </span>
                                </div>
                                <a data-behaviour="ga-event-synonyms"
                                    data-value="view synonyms" href="/en/synonym/cold">View synonyms
                                </a>
                            </div>
                        </div>
                        <ol class="subSenses">
                            ....
                            ....
                        </ol>
                    </div>
                </li>
                <li>
                    (Com a mesma estrutura do li acima, variando a informacao sobre o significado e os sinonimos)
                </li>
            </section>
            (contem varias seccoes com class gramb, cada uma para diferentes significados semanticos da palavra)
```

Listing 6: Página HTML fornecida pelo Lexico.

Após efetuada uma análise ao *HTML* implementamos o seguinte *workflow* utilizando o *Beautiful Soup* para retirar a informação que necessitamos.

```
1
2       # Fazer o pedido com o URL previamente construido
3       response = requests.get(url).content
4
5       # Analise da pagina html pelo Beautiful Soup
6       soup = BS(response, 'html.parser')
7
8       # Encontrar todas as seccoes gramb
9       word_types = soup.findAll('section', 'gramb')
10
11      # Por cada seccao encontrada
12      for word_type in word_types:
13
14          # Encontrar o significado semantico da palavra na section
15          instance = word_type.find('h3', 'ps pos')
16
17          # Encontrar a lista na section que contem todos os significados e sinonimos
18          ul = word_type.find('ul', 'semb')
19
20          # Encontrar todos os lis que essa lista contem
21          lis = ul.findAll('li')
22
23          # Por cada li encontrado
24          for li in lis:
25
26              # Encontrar a DIV com classe trg
```

```
            div = li.find('div', 'trg')

            # Se a DIV existir
            if(div):
                # Encontrar o paragrafo onde o significado da palavra esta
                p = div.find('p')

                # Se existir
                if(p):
                    # Retirar o significado que se encontra dentro do span
                    span = p.find('span', 'ind')

                    # Encontrar a DIV que contem os sinonimos que está dentro da DIV trg
                    sysn = div.find('div', 'synonyms')

                    # Se existir
                    if(sysn):

                        # Retirar o primeiro sinonimo
                        strong = sysn.find('strong', 'syn')

                        # Retirar os restantes sinonimos
                        exg = sysn.find('span', 'syn')
```

Listing 7: Implementação com o Beautiful Soup para o *Lexico*.

É de salientar que para o mesmo significado semântico, possuímos vários significados e como consequência várias palavras sinónimas. Então para resolver isso, fazemos uma concatenação de significados, separando-os para o mesmo significado semântico pelos carateres *(///)*. O mesmo é aplicado para as palavras sinónimas.

5 Apresentação do Resultado Final

Para testar o nosso *script* precisamos de executar o seguinte comando.

```
./scriptname --to pt,it,fr cold
```

Este comando, irá gerar um ficheiro *TMX* com toda a informação descrita acima e com as traduções da palavra *cold* para português, italiano e francês.

```
<?xml version="1.0" encoding="UTF-8"?>
<tmx version="1.4">
  <header adminlang="en"
    datatype="tbx"
```

```
    o-tmf="unknown"
    segtype="block"
    srclang="en"/>
  <body>
    <tu tuid="1">
      <prop type="word_type">adjective</prop>
      <prop type="meaning">Of or at a low or relatively low
    temperature, especially when compared with the human body. ||
     Lacking affection or warmth of feeling; unemotional. || (of
    the scent or trail of a hunted person or animal) no longer
    fresh and easy to follow. || Without preparation or rehearsal
    .</prop>
      <prop type="synonyms">chilly, cool, freezing, icy, snowy,
    icy-cold, glacial, wintry, crisp, frosty, frigid, bitter,
    bitterly cold, biting, piercing, numbing, sharp, raw, polar,
    arctic, Siberian || unfriendly, cool, inhospitable,
    unwelcoming, unsympathetic, forbidding, stony, frigid, frosty
    , glacial, lukewarm, haughty, supercilious, disdainful, aloof
    , distant, remote, indifferent, reserved, withdrawn,
    uncommunicative, unresponsive, unfeeling, unemotional,
    dispassionate, passionless, wooden, impersonal, formal, stiff
    , austere || Doesn't have synonyms || unprepared, unready,
    inattentive, unwary, unwatchful, with one's defences down, by
     surprise, cold, unsuspecting</prop>
      <tuv xml:lang="en">
        <seg>cold</seg>
      </tuv>
      <tuv xml:lang="fr">
        <seg>froide</seg>
      </tuv>
      <tuv xml:lang="pt">
        <seg>frio</seg>
      </tuv>
      <tuv xml:lang="it">
        <seg>fredde</seg>
      </tuv>
    </tu>


    <tu tuid="2">
      <prop type="word_type">noun</prop>
      <prop type="meaning">A low temperature; cold weather; a
    cold environment. || A common infection in which the mucous
    membrane of the nose and throat becomes inflamed, typically
    causing running at the nose, sneezing, and a sore throat.</
    prop>
      <prop type="synonyms">Doesn't have synonyms || cold, dose
    of flu, dose of influenza, respiratory infection, viral
    infection, virus</prop>
```

```
32        <tuv xml:lang="en">
33          <seg>cold</seg>
34        </tuv>
35        <tuv xml:lang="fr">
36          <seg>rhume</seg>
37        </tuv>
38        <tuv xml:lang="pt">
39          <seg>constipação</seg>
40        </tuv>
41        <tuv xml:lang="it">
42          <seg>freddo</seg>
43        </tuv>
44      </tu>
45    </body>
46  </tmx>
```

Listing 8: Ficheiro resultando do comando.

6 Conclusão

A realização deste trabalho prático foi importante para o nosso desenvolvimento como futuros engenheiros informáticos, pois permitiu-nos aprofundar os conhecimentos sobre *web scraping* e sobre ferramentas que o auxiliam como o *Beautiful Soup*. Para além disso, apresentou-nos um formato de ficheiro que nunca tínhamos ouvido falar e que viemos a descobrir ser muito importante no âmbito das traduções.

Por fim, podemos concluir que o *Beautiful Soup* é uma ferramenta poderosa, capaz de nos ajudar na busca de informação necessária para a construção de um projeto de maneira eficaz e direta.

7 Referências

- Lista de Elementos do *TMX*
- Lista de Atributos do *TMX*
- *Linguee*
- *Lexico*