Automatizace výstavby ontologie s využitím slovníku a báze dokumentů
...
terms

Vojtěch Ondryhal, Ladislav Buřita

27. března 2013

Obsah

1 Úvod

Dokument podrobně popisuje vlastnosti a použití nástroje pro analýzu obsahu dokumentu na základě existující ontologie. Nástroj je obecný, lze jej aplikovat na libovolném systému, který je schopen zpřístupnit svoji dokumentovou bázi a poskytnout instance jednotlivých témat ontologie. Nástroj byl vyzkoušen na projektu Lessons Learned, který je implementován v prostředí AToM2 společnosti AION CS s.r.o.

Nástroj je schopen vyhledat jednotlivé termíny, jejich varianty či zkratky a připravit vstupní dávku pro import nalezených asociací.

2 Použití

Projekt lze použít ve formě zdrojového kódu v jazyce Python na platformách Linux, Windows, Mac, popř. lze připravit spustitelnou distribuci připravenou pro prostředí Windows.

2.1 Python

```
python terms.py source
```

parametr `source` určuje téma v ontologii, pro které bude provedena analýza. Implicitně nástroj předpokládá, že existuje vstupní soubor `source-terms.txt` ve formátu CSV

(Comma Separated Values), kde první termín odpovídá názvu instance a ostatní jsou jeho varianty či zkratky.

Např. `organizace-terms.txt` obsahuje instance tématu Organizace v ontologii.

```
Společné velitelství NATO,NATO HQ,NATOs headquarters
Tálibán,Taleban
tým CIMIC,CIMIC,Civil-Military Co-operation, civilně vojenská spolupráce
tým EOD,EOD,Explosive Ordnance Disposal, pyrotechnické očištění území
tým HUMINT,HUMINT,HUMan INTeligence, špionáž, kontrašpionáž, sběr informací
tým PSYOPS,PSYOPS,Psychologická operace, Psychological operation
úkolové uskupení,ÚO,task force, TF
US Army,United States Armed Forces, Ozbrojené síly USA
```

3 Případová studie použití - Lessons Learned

V systému AtoM2 je uloženo 2070 článků k projektu Lessons Learned. Máme-li připravenou dokumentovou bázi, pak lze např. pro téma proces provést vyhodnocení s následujícím výsledkem:

```
(START)
Py Version: 3.3.0
Searching for occurences ...
Loading articles ...
... loaded: 2070
Loading terms ...
... loaded: 26
Analyzing ...
... total associations: 380
Exporting ...
(END)
```

Výsledek je uložen do souboru `proces-assocs.txt` a je připraven pro import do cílové aplikace, v tomto případě AtoM2.

```
PROCES;Eskalace síly;c040201 Definice používané pro pojem ESKALACE SÍLY
PROCES;Pátrání;c1105010109 POZOR
PROCES;Útok;c19030701 Palebná podpora
PROCES;Přesun;c1003010404 Mobilita
PROCES;Léčka;c160601 Provedení dokladu pro potřeby hlídky
PROCES;Útok;c160601 Provedení dokladu pro potřeby hlídky
PROCES;Eskalace síly;c04040305 Grafické pomůcky pro vojáky
PROCES;Přesun;c12050404 Hlídky na vozidlech
PROCES;Léčka;c0602010101 Cíle léčky
```

4 Příloha A

V příloze A je uveden klíčový zdrojový kód pro vlastní analýzu dokumentové báze. Jedná se o třídu **TermsAnalysis**.

```
# Document analysis

import os
import re
import sys

class TermsAnalysis:

  dir = None
  source_name = None
  articles = {}
  terms_fname = None
  terms = {}
  analysis = []

  def __init__(self, dir, source_name):
    """Inicialization"""
    self.dir = dir
    self.source_name = source_name
    self.terms_fname = source_name + '-terms.txt'

  def load_articles(self):
    """Loads files into dictionary"""
    filenames = os.listdir(self.dir)
    for filename in filenames:
      path = os.path.join(self.dir, filename)
      file = open(path, 'r', encoding = 'cp1250')
      self.articles['c' + os.path.splitext(filename)[0]] =
          file.read()
    print ('... loaded: ' + str(len(filenames)))

  def load_terms(self):
    """Documentation"""
    file = open(self.terms_fname, 'r', encoding = 'utf-8')
    lines = file.readlines()
    file.close()
    for line in lines:
      line = line.strip()
      if len(line) > 0:
        items = line.split(',')
        # Remove whitespaces
        new_items = []
        for item in items:
          new_item = item.strip()
```

```
            new_items.append(new_item)
          self.terms[new_items[0]] = new_items
      print ('... loaded: ' + str(len(lines)))

    def analyze(self):
      """Analyze dictionaty articles with terms"""
      for article_name in self.articles.keys():
        article = self.articles[article_name]
        for term in self.terms.keys():
          for item in self.terms[term]:
            match = re.search(item, article)
            if match:
              self.analysis.append((term, article_name))
      print ('... total associations: ' + str(len(self.analysis
        )))

    def export(self):
      """Export into txt"""
      file = open(self.source_name + '-assocs.txt', 'w',
        encoding='utf-8')
      for tuple in self.analysis:
        file.write(self.source_name.upper() + ';' + tuple[0] +
          ';' + tuple[1] + '\n')
      file.close()


  def main():
    source_name = sys.argv[1]
    print ("(START)")
    print ('Py Version: ' + sys.version.split(' ')[0])
    print ("Searching for occurences ...")
    analysis = TermsAnalysis('articles', source_name)
    print ("Loading articles ...")
    analysis.load_articles()
    print ("Loading terms ...")
    analysis.load_terms()
    print ("Analyzing ...")
    analysis.analyze()
    print ("Exporting ...")
    analysis.export()
    print ("(END)")

  if __name__ == "__main__":
    main()
```

5 Příloha B

Příklad exportovacího skriptu pro tvorbu slovníku vstpuních termínů ze znalostního systému.

K dispozici je va výstupu systému následující zdroj. Tento je převeden do požadovaného formátu vytvořené aplikace.

```
<Row>
 <Cell Index="0">PROCES</Cell>
 <Cell Index="1">Protipovstalecke operace</Cell>
 <Cell Index="2">COIN</Cell>
 <Cell Index="3">Counter insurgency</Cell>
 </Row>
```

Následuje zdrojový kód:

```
# Export from AtoM2
__author__="ondryhalv"
__date__ ="$18.6.2012 13:29:40$"

import sys
import re

def parse(source_name):
  print('parsing started ....')
  filename = "data-in\\" + source_name + '.xml'
  file = open(filename, 'r')
  file = open(filename, 'r', encoding='utf-8')
  text = file.read()
  regex_class = '<Row>(.*?)</Row>'
  rows = re.findall(regex_class, text)

  target = open(source_name + '-terms.txt', 'w', encoding='
     utf-8')
  for row in rows:
        regex_class = '<Cell Index="[123]">(.*?)</Cell>'
        cells = re.findall(regex_class, row)
        print(cells)
        line = ",".join(cells)
        target.write(line)
        target.write('\n')
  target.close()

# Parameter is a source - Topic in knowledge Base
def main():
        parse(sys.argv[1])

if __name__ == "__main__":
  main()
```