PyMedTermino:
an open-source generic API
for advanced terminology services

Jean-Baptiste LAMY
Alain VENOT
Catherine DUCLOS

LIMICS, Université Paris 13, Sorbonne Paris Cité, Université Paris 6,
INSERM UMR_S 1142, 74 rue Marcel Cachin, 93017 Bobigny, France

MIE 2015 – Madrid 5/2015
Introduction

- Terminological resources play a crucial role in medical informatics research and software applications but...

- Heterogeneity between terminologies:
  - Monoaxial vs multiaxial
  - Single vs multiple language
  - Pre- vs post-coordinated
  - Textual vs graphical

Child game
Introduction

- **PyMedTermino**: a generic API for a multi-terminology multilingual terminology service
- Mainly for research and educational purposes
  - Batch processing of terminologies
  - Advanced terminological operations
- Implemented in the Python programming language
- Support 5 terminological resources + UMLS
- “Write once, code with every terminologies”

- Based on the terminological services that were developed for managing VCM icons (Visualization of Concept in Medicine)
  - VCM icons are post-coordinated
  - Mappings with other terminologies
Material: the VCM iconic language

A compositional language

- Color is for the temporal aspect (past, current or risk of disorder)
- Central pictogram is for anatomico-functional location
- Shape modifiers are for generic pathological processes and transversal anatomical structures

http://vcm.univ-paris13.fr/
Materials: terminological resources

- **ICD10**
  International Classification of Diseases
  version 10-2010 (English, French)

- **SNOMED CT**
  Systematized NOmenclature of MEDicine
  Clinical Terms
  version 2014-01-31 (English)

- **UMLS**
  Unified Medical Language System
  version 2012AA (Multilingual)

- **VCM icons**
  Visualization of Concept in Medicine
  version 2014 (Graphical, post-coordinated)

- **MedDRA**
  Medical Dictionary for Regulatory Activities
  version 17.1 (Multilingual)

- **CDF**
  CoDiFication from the Thériaque drug databank
  version 2014 (French)

Used for designing the generic model

Added after the design of the generic model for validation
## Basic and derived operations in the generic model

<table>
<thead>
<tr>
<th>On a terminology</th>
<th>Iterate over all concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Obtain the first level concepts</td>
</tr>
<tr>
<td></td>
<td>Obtain a concept from a code</td>
</tr>
<tr>
<td></td>
<td>Free-text search</td>
</tr>
<tr>
<td>On a concept</td>
<td>Obtain the code of the concept</td>
</tr>
<tr>
<td></td>
<td>Obtain the preferred term in a given language</td>
</tr>
<tr>
<td></td>
<td>Obtain all the terms in a given language</td>
</tr>
<tr>
<td></td>
<td>Obtain parent concepts</td>
</tr>
<tr>
<td></td>
<td>Obtain children concepts</td>
</tr>
<tr>
<td></td>
<td>Obtain the lists of available (non is-a) relations</td>
</tr>
<tr>
<td></td>
<td>Obtain the values of a given relation</td>
</tr>
<tr>
<td></td>
<td>Test if a concept is a descendant of another concept</td>
</tr>
<tr>
<td></td>
<td>Iterate over ancestor concepts, with or without doubles</td>
</tr>
<tr>
<td></td>
<td>Iterate over descendant concepts, with or without doubles</td>
</tr>
<tr>
<td>On a set of concepts</td>
<td>Find all concepts in the set that are, or are not, another concept</td>
</tr>
<tr>
<td></td>
<td>Keep only the most generic or the most specific concepts in the set</td>
</tr>
<tr>
<td></td>
<td>Compute the lowest common ancestors</td>
</tr>
<tr>
<td></td>
<td>Test if set A is a semantic subset of set B</td>
</tr>
<tr>
<td></td>
<td>Perform usual set operations (union, intersection, difference, etc)</td>
</tr>
<tr>
<td>On a mapping</td>
<td>Map a concept to another terminology</td>
</tr>
<tr>
<td></td>
<td>Map a set of concepts</td>
</tr>
<tr>
<td></td>
<td>Create the reverse mapping</td>
</tr>
<tr>
<td></td>
<td>Chain the mapping to another mapping, resulting in a new mapping</td>
</tr>
</tbody>
</table>
**Basic and derived operations in the generic model**

**On a terminology**
- Iterate over all concepts
- Obtain the first level concepts
- Obtain a concept from a code
- Free-text search

**On a concept**
- Obtain the code of the concept
- Obtain the preferred term in a given language
- Obtain all the terms in a given language
- **Obtain parent concepts**
- Obtain children concepts
- Obtain the lists of available (non is-a) relations
- Obtain the values of a given relation
- Test if a concept is a descendant of another concept
- **Iterate over ancestor concepts, with or without doubles**
- Iterate over descendant concepts, with or without doubles

**On a set of concepts**
- Find all concepts in the set that are, or are not, another concept
- Keep only the most generic or the most specific concepts in the set
- **Compute the lowest common ancestors**
- Test if set A is a semantic subset of set B
- Perform usual set operations (union, intersection, difference, etc)

**On a mapping**
- Map a concept to another terminology
- Map a set of concepts
- Create the reverse mapping
- Chain the mapping to another mapping, resulting in a new mapping
General architecture

- No big database (e.g. UMLS): SQL is not sufficient for terminology (no recursion)
- **Basic operations**, implemented for each terminology using:
  - A terminology-specific database, or
  - Post-coordination algorithms
- **Derived operations**, whose implementation depends on the terminology structure
Generic model

- Five main classes:
  - Mapping
  - Terminology
  - Concept
  - Term
  - Concepts (set of concepts)
Examples of use

- Command-line interface or program
- Search SNOMED CT for *ulcer of duodenum*:

```python
>>> SNOMEDCT.search("ulcer duoden*")
[Snomed6761005]  # Familial hypergastrinemic duodenal ulcer (disorder)
, Snomed12355008 # Duodenal ulcer with hemorrhage, with perforation AND with obstruction (disorder)
, Snomed12847006 # Acute duodenal ulcer with hemorrhage (disorder)
, Snomed15115006 # Duodenal ulcer with hemorrhage AND with perforation but without obstruction (disorder)
, Snomed16516008 # Familial duodenal ulcer associated with rapid gastric emptying (disorder)
, Snomed18169007 # Duodenal ulcer without hemorrhage AND without perforation but with obstruction (disorder)
... ] # 71 results :-(
```
Examples of use

- Search and use the “keep most generic” derived operation:

```python
>>> concepts = Concepts(SNOMEDCT.search("ulcer duoden*"))
>>> concepts.keep_most_generic()
>>> concepts
Concepts([SNOMEDCT[275127005] # Family history: Duodenal ulcer,
        SNOMEDCT[275547005] # History of duodenal ulcer,
        SNOMEDCT[473216000] # Suture plication of artery for control of duodenal ulcer hemorrhage (procedure),
        SNOMEDCT[314627002] # Endoscopic injection hemostasis of duodenal ulcer (procedure),
        SNOMEDCT[413213005] # Ulcerogenic deformed duodenum,
        SNOMEDCT[173887007] # Duodenal ulcer operation,
        SNOMEDCT[51868009] # Duodenal ulcer disease (disorder)]) # 7 results :)```
Examples of use

- Mapping example:

  ```
  >>> SNOMEDCT[51868009] >> VCM
  Concepts([VCM[u"current--lesion--intestine"] # ulcer of intestine])
  ```

- Chaining mappings:

  ```
  >>> ICD10["I10"] >> SNOMEDCT >> VCM
  Concepts([VCM[u"current--hyper-vessel_bp"] # hypertension])
  ```
Examples of use

Search for all **clinical findings** (id 404684003) in SNOMED CT

    with “hemorrhag” in their term

    but not associated with the **hemorrhage morphology** (id 50960005)

Script:
```
from pymedtermino.snomedct import *
for concept in SNOMEDCT.search("hemorrhag"):  
    if not concept.is_a(SNOMEDCT[404684003]): continue
    has_hemorrhage = False
    for hemorrhage in SNOMEDCT[50960005].self_and_descendants_no_double():
        if hemorrhage in concept.associated_morphology:  
            has_hemorrhage = True
            break
    if not has_hemorrhage: print(concept)
```

Output:
```
SNOMEDCT[37442009]  # Peptic ulcer without hemorrhage AND without perforation (disorder)
SNOMEDCT[240523007]  # Viral hemorrhagic fever (disorder)
... (154 concepts listed)
```
Discussion

- PyMedTermino has been used in research projects:
  - **VCM** (iconic terminology)
  - **SiFaDo** (tools for facilitating medical coding, ANR)
  - **VIIIP** (comparison of new drugs with older ones, ANSM)

- The advanced terminological operations we proposed were useful
  - *e.g.* keep most generic, operations on a set of concepts

- PyMedTermino has been used in training sessions with students in master of biomedical informatics (M1 and M2)
  - Students can compare ICD10 and SNOMED CT
  - Technically more interesting than navigating in a terminology browser

- Main limits: quality of UMLS mappings, available only for Python
Discussion

- In the literature, Most terminological services
  - propose similar basic operations [Pathak J]
  - are aimed at browsing terminologies, such as Hetop [Grosjean J]
  - or at hospital use (rather than research and education)

- PyMedTermino is a Free Software (GNU LGPL license)
  - [http://pypi.python.org/pypi/PyMedTermino](http://pypi.python.org/pypi/PyMedTermino)
  - Terminology contents are not included (due to copyright)
  - But PyMedTermino includes:
    - Links for free downloads of terminologies (ICD10, SNOMED CT, etc)
    - Scripts for converting terminologies into optimized SQLite databases
    - The generic API previously described, built over these databases
References


