termate Documentation

Release 0.1.5

De Nederlandsche Bank

Jan 22, 2023

CONTENTS:

CHAPTER

ONE

TERMATE

Your Python companion for using termbases in NLP analyses

Termate is a Python package for terminology management with the TermBase eXchange (TBX) format

DISCLAIMER - BETA PHASE

This package is currently in a beta phase.

- Free software: MIT license

Documentation can be found here here

1.1 Main features

- Using existing TBX termbases in NLP analyses (TBX version 3)
- Creating TBX termbases based on extracted terms from documents
- Converting TBX termbases to Semantic Web data (OntoLex Lemon)

Termate works together with nafigator.

CHAPTER

TWO

INSTALLATION

2.1 Stable release

To install termate, run this command in your terminal:

```
$ pip install termate
```

This is the preferred method to install termate, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for termate can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/DeNederlandscheBank/termate
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER

THREE

AVAILABLE TERMBASES

A terminology database (termbase) forms a structured way to store and exchange dictionaries and lexicographical data of related terms. Termate provides a way to use these termbases in NLP analyses.

3.1 The IATE termbase

The IATE termbase can be found here

- Interactive Terminology for Europe.

A conceptEntry looks like this:

```
<conceptEntry id="iate_3515206">
  <descrip type="subjectField">insurance</descrip>
  <langSec xml:lang="en">
    <termSec>
      <term>risk margin</term>
      <termNote type="termType">fullForm</termNote>
      <descrip type="reliabilityCode">9</descrip>
    </termSec>
  </langSec>
  <langSec xml:lang="fi">
    <termSec>
      <term>riskimarginaali</term>
      <termNote type="termType">fullForm</termNote>
      <descrip type="reliabilityCode">9</descrip>
    </termSec>
  </langSec>
  <langSec xml:lang="fr">
    <termSec>
      <term>marge de risque</term>
      <termNote type="termType">fullForm</termNote>
      <descrip type="reliabilityCode">9</descrip>
    </termSec>
  </langSec>
  ...
  </conceptEntry>
```

3.2 The Supervisory termbases

Specific termbases are published on data.world in the Termbase repository. They include:

- EIOPA Solvency 2 XBRL Taxonomy, version 2.6.0
- EBA CRD XBRL Taxonomy, version 3.2.1.0

These termbases combine three data sources: terms from a supervisory XBRL Taxonomy (derived from the labels of XBRL elements), terms from the IATE termbase that match the labels of the XBRL elements (available in all official European languages) and language-specific linguistic annotations to each term from an NLP processor.

References to XBRL elements are added as cross references directly under the conceptEntry (with match type fullMatch and partialMatch to specify whether the term is an exact match with a label of the XBRL element or that the term is a substring of the label).

Two linguistic annotations are added to each term in the termbase: the lemma and the part of speech tags. The linguistic annotations are added as termNotes in the term section of a term (with type termLemma and partOfSpeech). If a term contains more than one word then the part of speech tags are in a comma-separated string.

```
<conceptEntry id="iate_3515206">
  <descrip type="subjectField">insurance</descrip>
  <langSec xml:lang="en">
    <termSec>
      <term>risk margin</term>
      <termNote type="termType">fullForm</termNote>
      <descrip type="reliabilityCode">9</descrip>
      <termNote type="termLemma">risk margin</termNote>
      <termNote type="partOfSpeech">noun, noun</termNote>
    </termSec>
  </langSec>
  <langSec xml:lang="fi">
    <termSec>
      <term>riskimarginaali</term>
      <termNote type="termType">fullForm</termNote>
      <descrip type="reliabilityCode">9</descrip>
      <termNote type="termLemma">riski#marginaali</termNote>
      <termNote type="partOfSpeech">noun</termNote>
    </termSec>
  </langSec>
  <langSec xml:lang="fr">
    <termSec>
      <term>marge de risque</term>
      <termNote type="termType">fullForm</termNote>
      <descrip type="reliabilityCode">9</descrip>
      <termNote type="termLemma">marge de risque</termNote>
      <termNote type="partOfSpeech">noun, adp, noun</termNote>
    </termSec>
  </langSec>
  ...
  <ref type="crossReference" match="fullMatch">http://eiopa.europa.eu/xbrl/s2c/dict/
↪dom/vm#x47</ref>
  <ref type="crossReference" match="fullMatch">http://eiopa.europa.eu/xbrl/s2md/fws/
↪solvency/solvency2/2021-07-15/tab/s.02.01.01.01#s2md_c653</ref>
  <ref type="crossReference" match="partialMatch">http://eiopa.europa.eu/xbrl/s2md/
↪fws/solvency/solvency2/2021-07-15/tab/s.26.06.01.01#s2md_c6792</ref>
    ...
  </conceptEntry>
```

For terms that are included in the XBRL Taxonomy for which no match could be found in the IATE database new conceptEntries were added. For example the term "valuation of recoverables":

```
<conceptEntry id="eiopa_23">
  <ref type="crossReference" match="fullMatch">http://eiopa.europa.eu/xbrl/s2c/dict/
↪dim#rr</ref>
  <langSec xml:lang="en">
    <termSec>
      <term>Valuation of recoverables</term>
      <termNote type="termType">fullForm</termNote>
      <termNote type="termLemma">valuation of recoverable</termNote>
      <termNote type="partOfSpeech">noun, adp, noun</termNote>
    </termSec>
  </langSec>
/conceptEntry>
```

This term is only available in the language of the XBRL Taxonomy. If translations are available then they can be included in the termbase by adding lines to the TBX Resource.

3.3 Using a termbase

If you have a TBX termbase available then you can read it in the following way:

```
IATE_FILE = os.path.join("..", "data", "termbases", "IATE_export.tbx")
termbase = termate.TbxDocument().open(IATE_FILE)
```

To get the concepts in the termbase as a list of dictionaries use:

```
concepts = termbase.concepts_list
```

The results of the first concept in the list then look for example like this:

```
{
'id': 'iate_127562',
'lang': {
  'en': [[
  {'type': 'term',
   'attr': {},
   'text': 'services agreement'
  },
  {'type': 'termNote',
   'attr': {'type': 'termType'},
   'text': 'fullForm'
  },
  {'type': 'descrip',
   'attr': {'type': 'reliabilityCode'},
   'text': '1'
  }
]]
...
```

3.4 Annotating NAF files

To annotate a NAF file with the content of a termbase open both files:

```
naf_file = "P:\\projects\\naf-data\\data\examples\\exmaple.naf.xml"
doc = nafigator.NafDocument().open(naf_file)

tbx_file = "P:\\projects\\tbx-data\\termbases\\EIOPA_SolvencyII_XBRL_Taxonomy_2.6.0_
↪PWD_with_External_Files.tbx"
termbase = termate.TbxDocument().open(tbx_file)
```

Then create a termbase processor and process with the processor the document:

```
t = nafigator.TermbaseProcessor(termbase)
t.process(doc=doc)
```

On initialization the TermbaseProcessor creates a fast way to access the terms in the termbase. After initialization you can process multiple documents with the same termbase by calling the process function.

Now you can overwrite the existing NAF file or store it under a different name

```
doc.write(naf_file)
```

CHAPTER

FOUR

TBX FORMAT

TBX, or TermBase eXchange, is an international standard for representing and exchanging information from termbases. TBX version 3 is published as ISO 30042:2019. A TBX Resource represents a collection of terminological concepts and is expressed as an XML file. It contains a header and a body of text with the terminological concepts. The main elements are described below.

4.1 Header

Header (tbxHeader): represents the metadata of the TBX Resource and contains the file description (fileDesc). The file description (fileDesc) contains (optional) title statement (titleStmt), publication statement (publicationStmt) and source description (sourceDesc).

```
<tbxHeader>
  <fileDesc>
    <sourceDesc>
      <p>EIOPA_SolvencyII_XBRL_Taxonomy_2.6.0_PWD_with_External_Files</p>
    </sourceDesc>
  </fileDesc>
</tbxHeader>
```

4.2 Concept entries

Terminological concept (conceptEntry): represents a language-independent concept. Each terminological concept has a unique ID, is described by a set of properties, such as the subject field it belongs to, and is associated to language sections, which are sets of language-specific terms that express the terminological concept.

Language section (langSec): a language section is a language-specific container for all terms that represent a terminological concept in a given language. The language section contains simple terms.

Term section (termSec): represents a language-specific term. A term section always contains a term with the text of the term and zero or more term notes (with term properties and linguistical properties) and descriptions (such as the reliability code of the term in relation to the concept). Related term notes are grouped in a term note group (termNoteGrp).

```
<conceptEntry id="iate_2149365">
  <descrip type="subjectField">insurance</descrip>
    <langSec xml:lang="en">
      <termSec>
        <term>risk mitigation</term>
        <termNote type="termType">fullForm</termNote>
```

(continues on next page)

(continued from previous page)

```
        <descrip type="reliabilityCode">9</descrip>
        <termNote type="termLemma">risk mitigation</termNote>
        <termNote type="partOfSpeech">noun, noun</termNote>
      </termSec>
    </langSec>
```

4.3 TBX Dialect

Version 3 of TBX provides dialect-specific schema to constrain TBX files. The TBX Resource contains the dialect name associated with a corresponding external schema. In this package a provisional private dialect TBX-DNB is used that extends the public dialect TBX-Basic with additional linguistic annotations.

- Introduction to TermBase eXchange (TBX) Version 3

CHAPTER

FIVE

TBX TO ONTOLEX LEMON

Termate provides an easy way to convert termbases to the OntoLex Lemon vocabulary. OntoLex Lemon is used to model lexicon and machine-readable dictionaries linked to the Semantic Web. The implementation in termate follows the W3C guidelines for converting TBX to RDF except that this implementation is based op TBX version 3.

Below you find examples from the converted Solvency 2 termbase.

5.1 Header information

The header of the termbase

```
@prefix dcat: <http://www.w3.org/ns/dcat#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix decomp: <http://www.w3.org/ns/lemon/decomp#> .
@prefix lexinfo: <http://www.lexinfo.net/ontology/3.0/lexinfo#> .
@prefix ontolex: <http://www.w3.org/ns/lemon/ontolex#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix tbx: <http://tbx2rdf.lider-project.eu/tbx#> .

<https://dnb.nl/rdf-data/termbases/solvency2/header> a tbx:Header,
        dcat:Dataset ;
    dcterms:type "TBX-DNB" ;
    tbx:encodingDesc "<p type=\"XCSURI\">TBXXCS.xcs</p>" ;
    tbx:sourceDesc "EIOPA_SolvencyII_XBRL_Taxonomy_2.6.0_PWD_with_External_Files" .
```

5.2 Terminological concepts

```
<https://dnb.nl/rdf-data/termbases/solvency2/iate_2149365> a skos:Concept ;
    tbx:subjectField "insurance" .
```

5.3 Lexical entries

The lexicon (only a few entries are shown):

```
<https://dnb.nl/rdf-data/termbases/solvency2/lexicon/en> a ontolex:Lexicon ;
    ontolex:entry <https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en>,
    <https://dnb.nl/rdf-data/termbases/solvency2/mitigation-en>,
    <https://dnb.nl/rdf-data/termbases/solvency2/risk-en> ;
    ontolex:language "en" .
```

The LexicalEntry of *risk mitigation*:

```
<https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en> a␣
↪ontolex:LexicalEntry,
        ontolex:MultiWordExpression ;
    rdfs:label "risk mitigation"@en ;
    tbx:reliabilityCode "9" ;
    tbx:termType "fullForm" ;
    decomp:constituent <https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en
↪#component1>,
        <https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en#component2> ;
    ontolex:canonicalForm <https://dnb.nl/rdf-data/termbases/solvency2/
↪risk+mitigation-en#CanonicalForm> ;
    ontolex:language "en" ;
    ontolex:sense <https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en
↪#Sense> .
```

The components:

```
<https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en#component1> a␣
↪decomp:Component ;
    decomp:correspondsTo <https://dnb.nl/rdf-data/termbases/solvency2/risk-en> .

<https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en#component2> a␣
↪decomp:Component ;
    decomp:correspondsTo <https://dnb.nl/rdf-data/termbases/solvency2/mitigation-en> .
```

The sense:

```
<https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en#Sense>␣
↪ontolex:reference <https://dnb.nl/rdf-data/termbases/solvency2/iate_2149365> .
```

And the other two LexicalEntries for *mitigation* and *risk*:

```
<https://dnb.nl/rdf-data/termbases/solvency2/risk-en> a ontolex:LexicalEntry,
        ontolex:Word ;
    rdfs:label "risk"@en ;
    lexinfo:partOfSpeech "noun" ;
    ontolex:language "en" .
```

```
<https://dnb.nl/rdf-data/termbases/solvency2/mitigation-en> a ontolex:LexicalEntry,
        ontolex:Word ;
    rdfs:label "mitigation"@en ;
    lexinfo:partOfSpeech "noun" ;
    ontolex:language "en" .
```

The CanonicalForm (the lemmatized version of the term):

```
<https://dnb.nl/rdf-data/termbases/solvency2/risk+mitigation-en#CanonicalForm> a␣
↪ontolex:Form ;
    ontolex:writtenRep "risk mitigation"@en .
```

CHAPTER

SIX

TERM EXTRACTION

6.1 Create a termbase from extracted terms

We generate an empty TBX document with

```
termbase = termate.TbxDocument()
termbase.generate(params = {
    termate.TBX_DIALECT: "TBX-DNB",
    termate.TBX_STYLE: "dca",
    termate.TBX_RELAXNG: "https://github.com/DeNederlandscheBank/termate/blob/main/
↪data/dialects/TBX-DNB.rng",
    termate.SOURCEDESC: ["TBX file, created via dnb/termate"],
    termate.TITLE: ["Example termbase"],
    termate.PUBLICATION: ["Created on ..."]
})
```

Then we extract terms from the Solvency II Delegated Acts (Dutch version) in NAF:

```
# create terms dictionary of subset of languages
terms = {}
for language in ['NL', 'EN', 'DE', 'FR', 'ES', 'ET', 'DA', 'SV']:
    DOC_FILE = "..\\..\\nafigator-data\\data\\legislation\\Solvency II Delegated Acts␣
↪- "+language+".naf.xml"
    doc = nafigator.NafDocument().open(DOC_FILE)
    termate.merge_terms_dict(terms, nafigator.extract_terms(doc))
```

Then we create a termbase

```
# add concepts from a dictionary of terms
termbase.create_tbx_from_terms_dict(terms=terms,
                                    params={'concept_id_prefix': 'tbx_'})
```

Then we add references from the InterActive Terminology for Europe (IATE) dataset:

```
# read the IATE file
IATE_FILE = "..//data//iate//IATE_export.tbx"
ref = termate.TbxDocument().open(IATE_FILE)
termbase.copy_from_tbx(reference=ref)
```

Then we add termnotes from the Dutch Lassy dataset (the small one) including basic insurance terms:

```
# read the lassy file
LASSY_FILE = "..//data//lassy//lassy_with_insurance.tbx"
```

(continues on next page)

(continued from previous page)

```
lassy = termate.TbxDocument().open(LASSY_FILE)
termbase.add_termnotes_from_tbx(reference=lassy, params={'number_of_word_components':␣
↪ 5})
```

Then we have a termbase with:

```
<conceptEntry id="249">
 <descrip type="subjectField">insurance</descrip>
 <xref>IATE_2246604</xref>
 <ref>https://iate.europa.eu/entry/result/2246604/en</ref>
 <langSec xml:lang="nl">
  <termSec>
   <term>solvabiliteitskapitaalvereiste</term>
   <termNote type="partOfSpeech">noun</termNote>
   <note>source: data/Solvency II Delegated Acts - NL.txt (#hits=331)</note>
   <termNote type="termType">fullForm</termNote>
   <descrip type="reliabilityCode">9</descrip>
   <termNote type="lemma">solvabiliteits_kapitaalvereiste</termNote>
   <termNote type="grammaticalNumber">singular</termNote>
   <termNoteGrp>
    <termNote type="component">solvabiliteits-</termNote>
    <termNote type="component">kapitaal-</termNote>
    <termNote type="component">vereiste</termNote>
   </termNoteGrp>
  </termSec>
 </langSec>
 <langSec xml:lang="en">
  <termSec>
   <term>SCR</term>
   <termNote type="termType">abbreviation</termNote>
   <descrip type="reliabilityCode">9</descrip>
  </termSec>
  <termSec>
   <term>solvency capital requirement</term>
   <termNote type="termType">fullForm</termNote>
   <descrip type="reliabilityCode">9</descrip>
   <termNote type="partOfSpeech">noun, noun, noun</termNote>
   <note>source: data/Solvency II Delegated Acts - EN.txt (#hits=266)</note>
  </termSec>
 </langSec>
 <langSec xml:lang="fr">
  <termSec>
   <term>capital de solvabilité requis</term>
   <termNote type="termType">fullForm</termNote>
   <descrip type="reliabilityCode">9</descrip>
   <termNote type="partOfSpeech">noun, adp, noun, adj</termNote>
   <note>source: ../nafigator-data/data/legislation/Solvency II Delegated Acts - FR.
↪txt (#hits=198)</note>
  </termSec>
  <termSec>
   <term>CSR</term>
   <termNote type="termType">abbreviation</termNote>
   <descrip type="reliabilityCode">9</descrip>
  </termSec>
 </langSec>
</conceptEntry>
```

- a reference is included to concept '2246604' from the IATE dataset. From that reference, we can for example derive that the official European term for this concept in English is 'solvency capital requirement' and in German 'Solvenzkapitalanforderung' and that the term is defined in Directive 2009/138/EC (Solvency II).
- termNotes include the partOfSpeech, lemma and morpohoFeats derived from the Lassy dataset (in Dutch). This dataset was extended with insurance related word components and terms that were not included in the Lassy dataset.
- also included are the word components of a term. The Dutch language, like the German language, often glues components together to construct new words instead of using separate words like the English language.

CHAPTER

SEVEN

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

7.1 Types of Contributions

7.1.1 Report Bugs

Report bugs at https://github.com/DeNederlandscheBank/termate/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

7.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

7.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

7.1.4 Write Documentation

termate could always use more documentation, whether as part of the official termate docs, in docstrings, or even on the web in blog posts, articles, and such.

7.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/DeNederlandscheBank/termate/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

7.2 Get Started!

Ready to contribute? Here's how to set up *termate* for local development.

1. Fork the *termate* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/termate.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv termate
$ cd termate/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 termate tests
$ python setup.py test or pytest
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

7.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.5, 3.6, 3.7 and 3.8, and for PyPy. Make sure that the tests pass for all supported Python versions.

7.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_termate
```

7.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER

EIGHT

CREDITS

8.1 Development Lead

- Willem Jan Willemse <w.j.willemse@dnb.nl>

8.2 Contributors

None yet. Why not be the first?

CHAPTER

NINE

HISTORY

9.1 0.1.0 (2022-01-02)

- First release on PyPI.

9.2 0.1.1 (2022-05-22)

- SKOS added.

9.3 0.1.3 (2022-08-07)

- Change to generate XBRL termbases

9.4 0.1.4 (2022-08-21)

- Improved documentation

9.5 0.1.5 (2022-08-23)

- Changed name of package to termate

CHAPTER

TEN

INDICES AND TABLES

- genindex
- modindex
- search