LIFT

Release 0.1

Francesca Giovannetti

Nov 03, 2021

CONTENTS

LIFT (Linked data from TEI) is a Python-based tool for transforming your TEI XML document into an RDF graph, ready for publication as linked open data on the web.

Note: This project is under active development.

CHAPTER

ONE

INTRODUCTION

LIFT (*Linked data from TEI*) is a Digital Humanities tool based on Python 2.7 for generating linked open data (LOD) out of TEI-encoded texts. This documentation is intended to provide you with an understanding of how LIFT works from the standpoint of both the technologies and the methodology employed. After reading this documentation, you should be able to use LIFT as is or modify its scripts to meet your specific project requirements.

LIFT offers five different TEI-to-LOD scripts written in Python 2.7. Each script addresses the transformation of a set of specific TEI entities (persons, persons and events, persons and relations, places, critical apparatus) to LOD. The scripts can be used independently from one another. An additional script comprehensive of all entities - except critical apparatuses (i.e. persons, events, relations, and places) - is also available. Another script for bibliographic entities is still under development.

This documentation, divided into four sections, serves a pedagogical purpose by demonstrating and discussing in detail a workflow for TEI-to-LOD transformation that does not require the use of XSLT and instead relies on Python (the lxml library is used for the processing of XML, while the RDFLib library is used for the creation of the RDF triples):

1. *Prepare your TEI document*, which explains the encoding requirements for the input TEI document;
2. *The RDF graph*, illustrating the structure and semantics of the generated knowledge graph by comparing the TEI input with the corresponding RDF output (this section also discusses why and how users might choose different ontologies for their projects or modify the scripts to work with different input data);
3. *How the scripts work*, where users can access an interactive Jupyter notebook containing LIFT's scripts with instructions and line-by-line explanations that are meant to help users develop the skills required to write their own transformation scripts;
4. *Further readings and resources*, which lists relevant publications and provides examples of TEI digital scholarly editions which have been enriched by means of linked open data.

You can directly run the scripts from the web application (https://projects.dharc.unibo.it/lift/quickstart.html) or download them for local use (see *How the scripts work* for more detailed instructions).

CHAPTER

TWO

PREPARE YOUR TEI DOCUMENT

The TEI standard allows for multiple ways to encode the same textual features. For example, you can use the tag <persName>, the tag <name>, or even the tag <rs> to markup a personal name (cf.).

This TEI feature has the advantage of flexibility, but it makes creating a universal TEI-to-RDF transformation script a difficult task. This is why this documentation includes a set of encoding guidelines designed to ensure a smooth TEI-to-RDF transformation via LIFT. In particular, in order for LIFT to work on your TEI document, you must adhere to the following simple guidelines:

1. *Provide TEI elements with unique identifiers using @xml:id*
2. *Provide an at least minimal TEI header*
3. *Use <person> and <persName> to represent persons and in-text references to such persons*
4. *Use <place> and <placeName> to represent places and in-text references to such places*
5. *Assign a @sameAs attribute to real-world entities*
6. *Encode relationships between persons within <listRelation>*
7. *Use <event> to represent events, either within <person> or <place>*

2.1 Provide TEI elements with unique identifiers using @xml:id

A unique URI must be assigned to each entity of a linked data graph (for example, a person, a place, a literary work, etc.). LIFT uses @xml:id attributes to create unique URIs. To accomplish this, LIFT concatenates the value of the attribute @xml:base attribute of the <TEI> element is concatenated with the value of the @xml:id attribute of the element. For example, the element below representing a person

```
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:base="https://example.org">
        ...
        <person xml:id="socr">...</person>
        ...
</TEI>
```

will be assigned the following URI:

```
<https://example.org/person/socr>
```

Note that the value of your @xml:base attribute should be registered as a permanent URL (i.e. through services such as w3id.org). Check the for more information on how to register your URL.

If your TEI document lacks unique identifiers, you can use Charlotte Tupman's (written for), which generates a unique identifier for each element that has no @xml:id. You can run the transformation using xsltproc after downloading the stylesheet to the same folder as your TEI document. You can look at this tutorial for detailed instructions about the process (last accessed 2021-10-25).

2.2 Provide an at least minimal TEI header

Your TEI header should comprise, at least, the minimal recommended elements as shown below:

```
<teiHeader>
        <fileDesc>
                <titleStmt>
                        <title><!-- Title of the resource --></title>
                        <author><!-- Author of the resource --></author>
                </titleStmt>
                <publicationStmt>
                        <p><!-- Information about the distribution of the resource --></
↪p>
                </publicationStmt>
                <sourceDesc>
                        <p><!-- Information about the source from which the resource␣
↪derives --></p>
                </sourceDesc>
        </fileDesc>
</teiHeader>
```

2.3 Use `<person>` and `<persName>` to represent persons and in-text references to such persons

Each person mentioned in the TEI document must be described in the TEI header within a `<person>` element to which an `@xml:id` has been assigned.

It is possible to provide a normalized form of each person's name by nesting a `<persName>` element containing the normalized name within `<person>`. You can provide multiple normalizations, e.g. in different languages (to specify the language use the `@xml:lang` attribute and a value from the of language codes).

All in-text occurrences of personal names must be encoded using `<persName>`. The attribute `@ref` should be used on the element to relate each name to the corresponding person (via the person's `@xml:id`). For example:

```
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:base="https://example.org">
        <teiHeader>
                ...
                <person xml:id="socr">
                        <persName xml:lang="en">Socrates</persName>
                        <persName xml:lang="el"></persName>
                </person>
                ...
        </teiHeader>
        <text>
                ...
                <persName ref="#socr">Socrates</persName>
                ...
```

(continues on next page)

(continued from previous page)

```
        </text>
</TEI>
```

Persons can be grouped using <listPerson>. Each <listPerson> (or, alternatively, each <person> element if <listPerson> is not present) can be assigned a @type and/or @corresp containing a short description of the group or individual. In particular, use @type for free-text descriptions (if using multi-word descriptions, please separate each word with an hyphen) or @corresp to provide a URI from a controlled vocabulary. For example:

```
<listPerson type="ancient-athenian-philosophers" corresp="http://dbpedia.org/class/yago/
↪WikicatAncientAthenianPhilosophers">
        <person xml:id="Socr">
        ...
```

2.4 Use <place> and <placeName> to represent places and in-text references to such places

The guidelines for encoding persons apply to places as well. For example:

```
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:base="https://example.org">
        <teiHeader>
        ...
                <place xml:id="athens">
                        <placeName xml:lang="en">Athens</placeName>
                </place>
        ...
        </teiHeader>
        <text>
        ...
                <placeName ref="#athens">Athens</persName>
        ...
        </text>
</TEI>
```

2.5 Assign a @sameAs attribute to real-world entities

By assigning a @sameAs attribute to your entities, you can disambiguate them by connecting them to external authority files, such as , , or the .

Provide a URI in a @sameAs attribute. You can supply multiple URIs, separated by a whitespace. For example:

```
<person xml:id="Socr" sameAs="http://viaf.org/viaf/88039167 http://id.loc.gov/rwo/agents/
↪n79055329">
```

2.6 Encode relationships between persons within <listRelation>

Use a series of <relation> elements nested within <listRelation> to markup relationships between persons in the TEI header. Note that <listRelation> must be a child element of <listPerson>.

In particular, for unidirectional relationships (e.g. 'Socrates has student Plato') use the attributes @active and @passive to express the subject and the object of the relationship respectively; for bidirectional relationships (e.g. 'Plato has colleague Xenophon') use the attribute @mutual. It is possible to represent a mutual relationship involving multiple persons by declaring more than one value for the @mutual attribute. Multiple values must be separated by whitespaces. Finally, use the @name attribute to express the nature of the relationship. You can reuse terms from , the Agent Relationship Ontology. For example:

```
<listRelation>
        <relation xml:id="rel01" name="hasStudent" active="#socr" passive="#plat #xen
↪#criti"/>
        <relation xml:id="rel02" name="hasColleague" mutual="#plat #xen"/>
</listRelation>
```

2.7 Use <event> to represent events, either within <person> or <place>

It is possible to describe events that occur in relation to a specific person or place. Such descriptions should be nested within the corresponding <person> or <place> elements.

The element <event> contains the description of the event. The attributes @type and @corresp can be assigned to <event> to provide a free-text label or a URI, respectively.

The date of the event must be recorded in @when or @from/@to attributes. Dates should be represented according to the standard.

A <label> can be used to provide a short textual description of the event, while a <desc> can contain the extended account of the event, including personal names, place names, and dates (encoded using the <date> element).

It is possible to specify the role held by the person in the event using the attribute @role and/or using the attribute @corresp on <persName>. The attribute @corresp should only contain a URI representing the role.

Furthermore, if there exist a primary or secondary source about the event, the element <bibl> can be used to express it (either as a child of <desc> or as a direct child of <event>). The <bibl> element may contain information about the <author>, the <title> and the <date> of publication of the source. A @sameAs can be associated to <bibl>.

For example:

```
<person xml:id="socr" sameAs="http://viaf.org/viaf/88039167">
        ...
        <event xml:id="ev01" type="trial" when="-0399" corresp="http://wordnet-rdf.
↪princeton.edu/id/01198357-n">
                <label>Socrates trial</label>
                <desc xml:id="desc01">The trial of <persName ref="#socr" role="defendant
↪" corresp="http://wordnet-rdf.princeton.edu/id/09781524-n">Socrates</persName> for␣
↪impiety and corruption of the youth took place in <placeName ref="#athens">Athens</
↪placeName> in <date when="-0399">399 B.C.</date></desc>
                <bibl xml:id="bibl01" sameAs="http://viaf.org/viaf/214045129"><author␣
↪ref="#plat">Plato</author> gives a contemporary account of the trial in his work␣
↪titled <title ref="Apology_of_Socr">Apology of Socrates</title>.</bibl>
        </event>
```

(continues on next page)

(continued from previous page)

```
        ...
</person>
```

2.8 Full example

You can download a TEI XML pseudo-edition featuring all of the examples presented above from .

CHAPTER

THREE

THE RDF GRAPH

LIFT takes a TEI document and returns an RDF graph. The semantics of the RDF graph is predefined, but you can modify it to suit your needs. For example, you could choose to reuse different ontologies or create new transformation rules.

The semantics of LIFT's RDF graphs is defined by the following ontologies:

- for representing persons and events
- for describing relationships between persons
- for basic information about some entities
- , an extension of CIDOC CRM, for representing texts
- for establishing links to external resources such as authority files
- for representing a person's role in the context of a specific event
- for linking a specific place to a role
- for specyfing the duration of a role in the context of an event
- for linking events to sources

This section compares the input TEI constructs with the corresponding RDF output to give you a better understanding of how LIFT creates your RDF triples. Please visit the above links to know more about each of the ontologies and properties reused in LIFT.

3.1 Persons

```xml
<?xml version="1.0" encoding="UTF-8"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:base="https://example.org" xml:id="example_
↪v1">
        <teiHeader>
                ...
                <listPerson type="ancient-athenian-philosophers" corresp="http://dbpedia.
↪org/class/yago/WikicatAncientAthenianPhilosophers">
                        <person xml:id="Socr" sameAs="http://viaf.org/viaf/88039167">
                                <persName xml:lang="en">Socrates</persName>
                        </person>
                </listPerson>
                ...
        </teiHeader>
        <text>
```

(continues on next page)

(continued from previous page)

```
                ...
                <p xml:id="para02">Some text mentioning <persName ref="#Aristot">
↪Aristotle</persName> and <placeName ref="#Sparta">Sparta</placeName> here.</p>
                ...
        </text>
</TEI>
```

```
<https://example.org/person/Aristot> a crm:E21_Person ;
        rdfs:label "Aristotle"@en ;
        owl:sameAs <http://dbpedia.org/resource/Aristotle>, <http://viaf.org/viaf/
↪7524651> ;
        dcterms:description "ancient athenian philosophers" ;
        dcterms:subject <http://dbpedia.org/class/yago/
↪WikicatAncientAthenianPhilosophers> ;
        dcterms:isReferencedBy <https://example.org/text/para02> .
```

3.2 Places

```
<?xml version="1.0" encoding="UTF-8"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:base="https://example.org" xml:id="example_
↪v1">
        <teiHeader>
                ...
                <listPlace>
                <place xml:id="Athens" sameAs="https://pleiades.stoa.org/places/579885">
                <placeName xml:lang="en">Athens</placeName>
                </place>
                ...
        </listPlace>
                ...
        </teiHeader>
        <text>
                ...
                <p xml:id="para01">Some text mentioning <persName ref="#Plat">Plato</
↪persName> and <placeName ref="#Athens">Athens</placeName>.</p>
                ...
        </text>
</TEI>
```

```
<https://example.org/place/Athens> a crm:E53_Place ;
        rdfs:label "Athens"@en ;
        owl:sameAs <https://pleiades.stoa.org/places/579885> ;
        dcterms:isReferencedBy <https://example.org/text/para01> .
```

3.3 Relations

```
<?xml version="1.0" encoding="UTF-8"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:base="https://example.org" xml:id="example_v1">
        <teiHeader>
                ...
                <listPerson>
                <listRelation>
                <relation xml:id="rel01" name="hasStudent" active="#Socr" passive="#Plat #Xen #Criti"/>
                <relation xml:id="rel02" name="hasColleague" mutual="#Plat #Xen"/>
                </listRelation>
                ...
        </listPerson>
                ...
        </teiHeader>
        ...
</TEI>
```

```
<https://example.org/person/Socr> a crm:E21_Person ;
        agrelon:hasStudent <https://example.org/person/Plat>, <https://example.org/person/Xen>, <https://example.org/person/Criti> .
```

```
<https://example.org/person/Plat> a crm:E21_Person ;
        agrelon:hasColleague <https://example.org/person/Xen> .
```

3.4 Events

```
<?xml version="1.0" encoding="UTF-8"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:base="https://example.org" xml:id="example_v1">
        <teiHeader>
                ...
                <listPerson type="ancient-athenian-philosophers" corresp="http://dbpedia.org/class/yago/WikicatAncientAthenianPhilosophers">
                        <person xml:id="Socr" sameAs="http://viaf.org/viaf/88039167">
                                ...
                                <event xml:id="ev01" type="trial" when="-0399" corresp="http://wordnet-rdf.princeton.edu/id/01198357-n">
                        <label>Socrates trial</label>
                        <desc xml:id="desc01">The trial of <persName ref="#Socr" role="defendant" corresp="http://wordnet-rdf.princeton.edu/id/09781524-n">Socrates</persName> for impiety and corruption of the youth took place in <placeName ref="#Athens">Athens</placeName> in <date when="-0399">399 B.C.</date></desc> <bibl xml:id="bibl01" sameAs="http://viaf.org/viaf/214045129"><author ref="#Plat">Plato</author> gives a contemporary account of the trial in his work titled <title ref="Apology_of_Socr">Apology of Socrates</title>.</bibl>
                </event>
```

(continues on next page)

(continued from previous page)

```
                        </person>
                </listPerson>
                ...
        </teiHeader>
        ...
</TEI>
```

```
<https://example.org/event/ev01> a crm:E5_Event ;
        rdfs:label "Socrates trial" ;
        dcterms:description "trial" ;
dcterms:subject <http://wordnet-rdf.princeton.edu/id/01198357-n> ;
prov:hadPrimarySource <https://example.org/source/bibl01> .

<https://example.org/source/bibl01> a prov:PrimarySource ;
        dcterms:creator <https://example.org/person/Plat> ;
        dcterms:title "Apology of Socrates" ;
        owl:sameAs <http://viaf.org/viaf/214045129> .
```

```
<https://example.org/person/Socr> a crm:E21_Person ;
        pro:holdsRoleInTime <https://example.org/Socr-in-ev01> .

<https://example.org/rit/Socr-at-ev01> a pro:RoleInTime ;
pro:relatesToEntity <https://example.org/event/ev01> ;
        pro:withRole <https://example.org/role/defendant> ;
        tvc:atTime <https://example.org/ev01-time> ;
        proles:relatesToPlace <https://example.org/place/Athens> .

<https://example.org/ev01-time> a <http://www.ontologydesignpatterns.org/cp/owl/
↪timeinterval.owl#TimeInterval> ;
        ti:hasIntervalEndDate "-0399"^^xsd:date ;
        ti:hasIntervalStartDate "-0399"^^xsd:date .

<https://example.org/role/defendant> a pro:Role ;
        rdfs:label "defendant" ;
        owl:sameAs <http://wordnet-rdf.princeton.edu/id/09781524-n> .
```

CHAPTER FOUR

HOW THE SCRIPTS WORK

LIFT features a set of five transformation scripts written in Python 2.7. LIFT leverages two libraries:

- lxml to find and select the relevant TEI constructs from the input XML document;
- RDFLib to create the RDF triples forming the knowledge graph.

Any of the scripts can be applied to your TEI document. If you use the scripts as is, make sure to read LIFT's encoding guidelines at *Prepare your TEI document* and update your input TEI document accordingly.

4.1 A Jupyter notebook walking through LIFT's TEI to RDF transformation line-by-line

The section *The RDF graph* displays the input TEI constructs next to the output RDF statements. A Jupyter notebook, available at this link, walks you through the scripts line-by-line.

You can read a non-interactive preview of the notebook by following the link above, or you can install Jupyer to access the notebook interactively. The second option requires a minimum familiarity with the command line (the Programming Historian provides excellent introductory tutorials for Windows as well as Mac/Linux users).

In order to access the notebook interactively

1. open Terminal or Prompt. If Python is already installed on your machine, run `pip install notebook` (visit https://jupyter.org/install for further help);
2. download the notebook;
3. in Terminal or Prompt navigate to the folder where the notebook was saved;
4. run `juptyter notebook` to open Jupyter on your browser;
5. from the browser, click on TEItoRDF.ipynb to access the notebook.

4.2 Modify the scripts and/or run them locally

After reading the notebook, you should be able to modify LIFT's scripts to meet the needs of your project. You can, for example, change how LIFT extracts information from the input file to avoid modifying your original TEI encoding, or you can enrich the knowledge graph with new RDF triples.

To modify LIFT the scripts and/or run them locally

1. go to LIFT's repository on Github and download the scripts;
2. open and change the scripts with an editor of your choice (remeber to update the path to the input TEI document (modify the line `tree = etree.parse('input.xml')` at the very beginning of the script);

3. **to run the transformation locally**

    1. open your Terminal or Prompt;
    2. navigate to the folder where the scripts are;
    3. run `python [name-of-your-script].py`.

CHAPTER

FIVE

FURTHER READINGS AND RESOURCES

Note: this section is updated on a regular basis

5.1 Tutorials

5.1.1 Understanding linked open data and the semantic web

- Tim Berners-Lee, "Linked Data", *Design Issues for the World Wide Web* (2006), https://www.w3.org/DesignIssues/LinkedData.html.
- Jonathan Blaney, "Introduction to the Principles of Linked Open Data," *The Programming Historian* 6 (2017), https://doi.org/10.46430/phen0068.

5.1.2 Using the command line

- (for Mac/Linux) Ian Milligan and James Baker, "Introduction to the Bash Command Line", *The Programming Historian* 3 (2014), https://doi.org/10.46430/phen0037.
- (for Windows) Ted Dawson, "Introduction to the Windows Command Line with PowerShell", *The Programming Historian* 5 (2016), https://doi.org/10.46430/phen0054.
- W3C Working Group Note, *RDF 1.1 Primer* (24 June 2014), https://www.w3.org/TR/rdf11-primer/.

5.1.3 Programming with Python for the Humanities

- William J. Turkel and Adam Crymble, "Python Introduction and Installation", *The Programming Historian* 1 (2012), https://doi.org/10.46430/phen0009.
- Silvio Peroni, *The CPT Book: A book for teaching Computational Thinking and Programming skills to people with a background in the Humanities* (last updated 2020), https://comp-think.github.io/.
- Folgert Karsdorp, Maarten van Gompel and Matt Munson, *Python Programming for the Humanities* (last updated 2017), https://www.karsdorp.io/python-course/.

5.2 Suggested readings

5.2.1 Linked data and the TEI

- Fabio Ciotti and Francesca Tomasi, "Formal Ontologies, Linked Data, and TEI Semantics", *Journal of the Text Encoding Initiative* 9 (2016), https://doi.org/10.4000/jtei.1480.
- Marilena Daquino, Francesca Giovannetti and Francesca Tomasi, "Linked Data per le edizioni scientifiche digitali. Il workflow di pubblicazione dell'edizione semantica del quaderno di appunti di Paolo Bufalini", *Umanistica Digitale* 7 (2019), https://doi.org/10.6092/issn.2532-8816/9091.
- Eide, Ø. "Ontologies, Data Modeling, and TEI", *Journal of the Text Encoding Initiative* 8 (2014). https://doi.org/10.4000/jtei.1191.

5.3 Useful resources

Pierre-Yves Vandenbussche, Bernard Vatant et al., *Linked Open Vocabularies (LOV): A gateway to reusable semantic vocabularies on the Web*, https://lov.linkeddata.es/dataset/lov/.

María Poveda-Villalón, Asunción Gómez-Pérez and Mari Carmen Suárez-Figueroa, "OOPS!(Ontology Pitfall Scanner!): An on-line tool for ontology evaluation", *International Journal on Semantic Web and Information Systems (IJSWIS)* 10.2 (2014): 7-34, http://oops.linkeddata.es/.