Transformation Approach of Open Web Data to Linked Open Data

Amina Meherhera[1], Imene Mekideche[1],
Leila Zemmouchi-Ghomari[2,*] and Abdessamed Réda Ghomari[3]

[1]*Ecole Nationale Supérieure d'Informatique, ESI, Algiers, Algeria*
[2]*Ecole Nationale Supérieure de Technologie, ENST, Algiers, Algeria*
[3]*LMCS Laboratory, Ecole Nationale Supérieure d'Informatique, ESI, Algiers, Algeria*
E-mail: fa_meherhera@esi.dz; fi_mekideche@esi.dz; leila.ghomari@enst.dz; a_ghomari@esi.dz
[*]*Corresponding Author*

Received 01 October 2020; Accepted 05 April 2021;
Publication 06 July 2021

Abstract

A large amount of data available over the Web and, in particular, the open data have, generally, heterogeneous formats and are not machine-readable. One promising solution to overcome the problems of heterogeneity and automatic interpretation is the Linked Data initiative, which aims to provide unified practices for publishing and contextually to link data on the Web, by using World Wide Web Consortium standards and the Semantic Web technologies. LinkedIn data promote the Web's transformation from a web of documents to a web of data, ensuring that machines and software agents can interpret the semantics of data correctly and therefore infer new facts and return relevant web data search results.

This paper presents an automatic generic transformation approach that manipulates several input formats of open web data to linked open data. This work aims to participate actively in the movement of publishing data compliant with linked data principles.

Keywords: Web data, open data, linked data, semantic web, transformation approaches.

Journal of Web Engineering, Vol. 20_5, 1247–1278.
doi: 10.13052/jwe1540-9589.2052

1 Introduction

Web data is often available in formats such as text files, scanned files, Excel sheets, and CSV files that can be processed only by humans, resulting in irrelevant search results that limit the inference of knowledge [1, 2].

To overcome this limitation, the inventor of the Web, Tim-Berners-Lee, set up the principles of the semantic Web in 2001 [3] and the linked data principles in 2007 [4] to enhance the use of standards and best practices that allow representing, structuring, and linking data on the Web [2]. These recommendations can significantly enhance discovery, accessibility, and exploitation of open web data [5] since they are machine-readable. The research questions that arise are:

- How to exploit datasets published on the Web in heterogeneous and human-readable formats?
- Moreover, how to extract data from documents and transform them into linked data that machines can use?

This work aims to propose a transformation approach supported by a tool of open web data in several human-readable formats to data compliant with the linked data principles to enable automatic processing of the resulting data.

This paper is structured as follows; first, we will provide an overview of background concepts, such as semantic Web, open data, and linked data. Second, we will describe related works and identify their limitations. The following section presents the proposed transformation approach. The next section is dedicated to the implementation and tests. Finally, we end with a conclusion, limitations, and directions for future action.

2 Background

This section defines the main concepts of the semantic Web, linked data, and related technologies.

2.1 Semantic Web

The semantic Web was designed as an extension of the classical Web that allows machines to search, combine intelligently, and process web content according to the meaning that this content has for humans. In the absence of artificial intelligence, this can only be accomplished if the desired meaning (i.e., semantics) of web resources is explicitly specified in a format that computers can process. For this, it is not enough to store the data in a

machine-processable syntax: a machine can process every HTML page on the Web, but it is also necessary that these data have formal semantics that specifies the conclusions to be drawn from the information collected [6].

Semantic Web and its technologies have been observed in many fields. They can organize and link data over the Web consistently and coherently. Semantic web technologies consist of RDF schema, OWL, and rule and query languages like SPARQL, and these technologies will help the various domains resolve their problems [7].

2.2 Open Data

Open data can be defined as[1]: "Open means **anyone** can **freely access**, **use**, **modify**, and **share** for **any** purpose (subject, at most, to requirements that preserve provenance and openness)." Alternatively, most succinctly: "Open data and content can **be freely used, modified, and shared by anyone** for any purpose."

Many organizations adopt Open Data Principles promising to make their open data complete, primary, and timely [8].

2.3 Linked Data

The term "Linked Data" refers to a set of best practices for publishing and interlinking structured data from different sources on the Web [9].

These best practices are known as linked data principles defined by Tim Berners Lee, the inventor of the Web and Semantic Web:

- Use URIs as names for things.
- Use HTTP URIs so that people can look up those names
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).
- Include links to other URIs so that they can discover more things.

Linked data transform the Web into a global database called the Web of data. Developers can query linked data from several sources at once and combine them on the fly, something challenging to do with traditional data management technologies [2].

Linked data is used beyond the LOD cloud and becomes the basis for data sharing in many contexts [10].

[1] http://opendefinition.org/

Figure 1 A graph representation of an RDF triple.

2.4 RDF

RDF (Resource Description Framework) is a data model for representing information about resources as a labeled oriented graph where a resource is described by several triples (subject - predicate -object), which reflect the basic structure of a simple sentence. An example of an RDF triple is:

Matt Briggs	has nickname	Matty
subject	predicate	Object

The same triplet may be represented as a graph, as shown in Figure 1.

2.5 SPARQL

SPARQL (SPARQL Protocol And RDF Query Language) is a set of specifications providing languages and protocols for querying and manipulating the content of RDF graphs on the Web or in an RDF Store.[2] Thus, SPARQL for RDF is like SQL for relational databases.

3 Related Works

Transforming open data to linked data has been widely studied in the literature. Different approaches with different architectures and file formats were proposed. The techniques used differ depending on the file format.

For tabular files, most approaches tend to map table components to RDF components: map a table row to RDF class (the subject), a table column to RDF property (predicate), and a table cell to RDF value (object). The difference between these approaches is the way this mapping is done: [11] creates RDF resources from table properties, while [12] and [13] create RDF

[2]https://www.w3.org/TR/2013/REC-sparql11-overview-20130321/, Dec 2019.

resources and then associate these resources with resources from existing vocabularies.

On the other hand, [14] and [15] map existing vocabularies directly to external resources.

A particular case is [16], which proposed to transform data provided in streaming from the AEMET dataset into linked data by developing a specific ontology for the dataset using predefined RDF templates and fill the data from the source files since the data format is static (predefined).

Some other approaches propose additional steps to improve the quality of the resulting data, for example [14] proposed to detect the relation between columns and form a new RDF class for better accuracy, while [11] has worked on algorithms to detect the type of cells (literal or URI object).

[13] addressed the problem from a different angle; it introduced an intermediate step: the transformation to the relational database before transforming into linked data.

As for text files, usually, extraction of text entities is performed using a NER (Named Entity Recognition) and relations between them and finally generate RDF triples. [17] used regular expressions to identify relations, while others generally prefer manual annotation.

The approaches proposed in the literature present different limitations:

- Many approaches are specific to a domain or a dataset and are hard to be adapted to other domains or datasets available on the Web.
- Some approaches focus on transforming only one file format (CSV format in most cases)
- Some approaches do not link resulting datasets to external datasets, thus not complying with the fourth linked data principle.
- Some approaches require human intervention (semi-automatic) and cannot be used by someone who does not know well-linked data concepts or the dataset to be converted.

4 Proposed Approach

Based on the limitations identified in the last section, we propose a new approach to transform open web data in CSV, text, HTML, pdf and image format to linked data that aims to overcome some issues revealed in the existing approaches.

In this section, we present the proposed approach (Figure 2). Next, we detail each step of the approach.

Table 1 The file formats supported by our approach with their corresponding MIME Type

Format	MIME Type
CSV	Text/csv
Text	Text/plain
HTML	Text/html
Image	image/png, image/jpeg
PDF	application/pdf

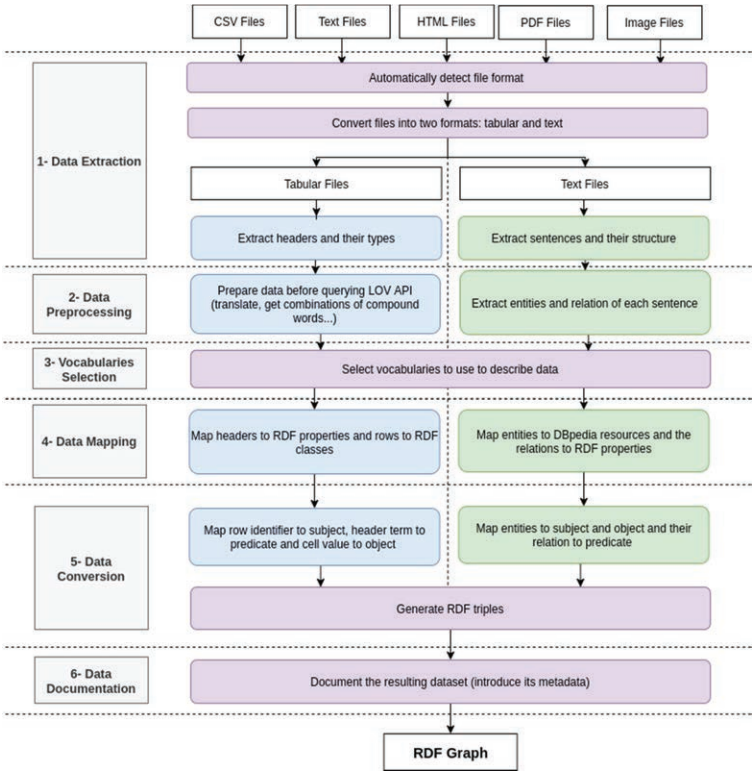


Figure 2 The workflow of the proposed approach.

4.1 Data Extraction

Since we will have to deal with different file formats, it would be better to convert them into two formats: tabular (CSV files) if the file contains tables and text if the file contains plain text.

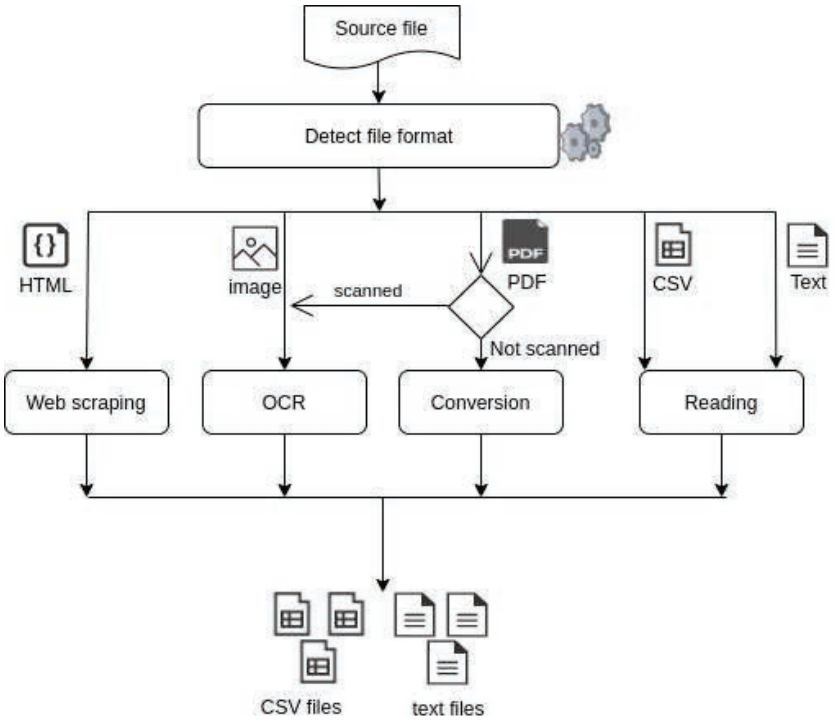


Figure 3 The techniques used to convert files to text files and tabular files.

The purpose of this step is to automatically detect the source file format and convert it either to CSV files, text files in both formats. Once the conversion is done, we will have to deal only with two formats instead of five formats, and we can extend the approach for other formats easily.

To perform this step, we have to perform these three actions:

a. Detect file format: using its MIME Type, a data format identifier included in the HTTP header 'content-type.' Table 1 represents formats supported by our approach with their MIME Types.

b. Convert the source file: Different techniques are used to convert all files to text or CSV format. The techniques used are represented in Figure 3.

If it is **a text or CSV** file, it consists of a simple reading of the file.

For **PDF files**, there are two use cases:

- **Scanned PDF files:** in this case, these files will be treated as images and therefore are converted using OCR (Optical Character Recognition),

which refers to computer processes for translating images of printed or typed text into text files.

- **Non-scanned PDF files:** in this case, tools are allowing to read the PDF file and extracting tables and texts, which is more efficient than an OCR.

For HTML files: HTML files contain useful data in text form. However, most web pages are not for ease of automated use. As a result, specialized tools and software have been developed to facilitate the scraping of web pages.

c. Extract data: once all files are converted to either tabular or text files (or both), we then extract data in these files: we extract table headers and their types from CSV files, and we extract sentences and grammatical structure from text files. Figure 4 represents the process of the data extraction step.

d.

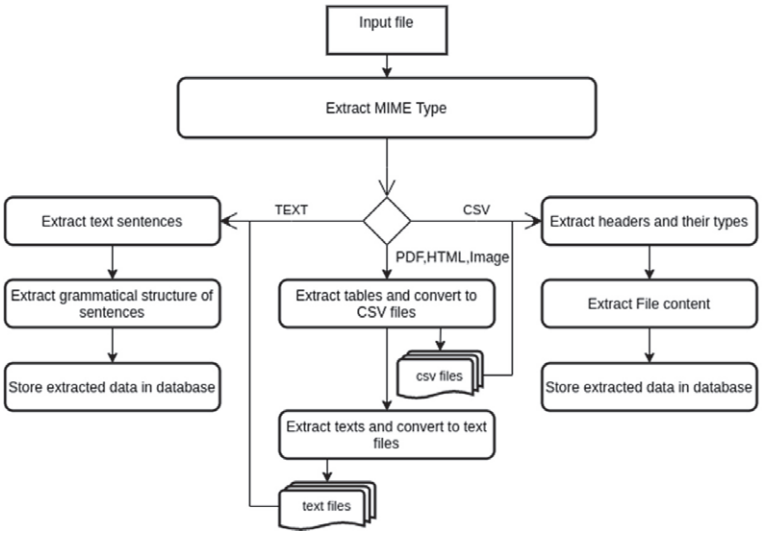


Figure 4 The process of the "Data Extraction" phase.

4.2 Data Preprocessing

In the proposed approach, in order to get RDF classes and properties from terms, we have chosen to use the LOV API (Linked Open Vocabularies),[3] where a list of more than 700 vocabularies is maintained by the LOV team

[3]https://lov.linkeddata.es/dataset/lov

of curators in charge of validating and inserting vocabularies in the LOV database and assigning them a detailed review (updated yearly).

LOV provides several API endpoints to:

- Get the list of LOV vocabularies.
- Search for a vocabulary based on its prefix or title
- Search for a term in LOV Vocabularies (class, property, instance, or datatype).

To query these endpoints and get relevant results, the data must respect a particular format.

The data processing phase aims to preprocess the data and prepare it before querying the LOV API.

4.2.1 Tabular files

We aim to process tabular file headers in a way that we get the most relevant results from LOV API:

a. Translation of a term: In the LOV vocabularies, all terms are expressed in English; for example, Figure 5 represents the search result for the word "prénom," and Figure 6 represents only the first 3 of 8 search results for the word "firstName." Hence the need to translate the terms into English before querying LOV.

To do this, we have used the googletrans library to translate the terms before searching them using the LOV API.

e. Have all possible combinations of a compound term: A term can be composed of several words and can thus be represented in different formats. If we take the example of "firstName," the different representations of the word are:

- Separated with space: first name.
- Attached: firstname.


```
[
  {
    "prefixedName": "dbpedia-owl:GivenName",
    "uri": "http://dbpedia.org/ontology/GivenName",
    "type": "class",
    "score": 2.3754375
  }
]
```


Figure 5 Search result for the term "prénom" in LOV.

```
{
  "prefixedName": "foaf:firstName",
  "uri": "http://xmlns.com/foaf/0.1/firstName",
  "type": "property",
  "score": 1
},
{
  "prefixedName": "ptop:firstName",
  "uri": "http://www.ontotext.com/proton/protontop#firstName",
  "type": "property",
  "score": 0.3264096
},
{
  "prefixedName": "swrc:firstName",
  "uri": "http://swrc.ontoware.org/ontology#firstName",
  "type": "property",
  "score": 0.32398805
},
```


Figure 6 The first three results of the search for the term “firstName” in LOV.


```
{
  "prefixedName": "sioc:first_name",
  "uri": "http://rdfs.org/sioc/ns#first_name",
  "type": "property",
  "score": 0.5555556
},
{
  "prefixedName": "sio:SIO_000181",
  "uri": "http://semanticscience.org/resource/SIO_000181",
  "type": "class",
  "score": 0.5555556
},
{
  "prefixedName": "ptop:firstName",
  "uri": "http://www.ontotext.com/proton/protontop#firstName",
  "type": "property",
  "score": 0.528172
},
```


Figure 7 The first three results of the search for the term “first name” in LOV.

- Separated with an underscore: first_name.
- Separated by a dash: first-name.
- Represented with camelCase format: firstName.

Different results are obtained for each of the above terms (see Figures from 7 to 10). On the other hand, we notice that the camelCase (Figure 10)

```
{
  "prefixedName": "foaf:firstName",
  "uri": "http://xmlns.com/foaf/0.1/firstName",
  "type": "property",
  "score": 1
},
{
  "prefixedName": "ptop:firstName",
  "uri": "http://www.ontotext.com/proton/protontop#firstName",
  "type": "property",
  "score": 0.3264096
},
{
  "prefixedName": "swrc:firstName",
  "uri": "http://swrc.ontoware.org/ontology#firstName",
  "type": "property",
  "score": 0.32398805
},
```


Figure 8 The first three results of the search for the term "firstname" in LOV.


```
[
  {
    "prefixedName": "sioc:first_name",
    "uri": "http://rdfs.org/sioc/ns#first_name",
    "type": "property",
    "score": 1.2676973
  }
]
```


Figure 9 Search results for the term "first_name" in LOV.

and attached (Figure 8) formats give the same results, and therefore, LOV is not sensitive to the case.

Table 2 is an example of the results of the construction of possible combinations.

4.2.2 Text files

To extract entities and relations from text files, we need to use an NLP (Natural Language Processing) service. Many tools and APIs are available in this domain (spacy, NLTK, IBM Watson, TextRazor…). After some testing and research, we chose to use TextRazor,[4] which allows extracting the entities present in the sentences and their relationships.

TextRazor's Relationship Extraction module uses a dependency analyzer and a set of sophisticated linguistic rules to analyze relationships in any type

[4]https://www.textrazor.com/, Mai 2020.

```
{
  "prefixedName": "foaf:firstName",
  "uri": "http://xmlns.com/foaf/0.1/firstName",
  "type": "property",
  "score": 1
},
{
  "prefixedName": "ptop:firstName",
  "uri": "http://www.ontotext.com/proton/protontop#firstName",
  "type": "property",
  "score": 0.3264096
},
{
  "prefixedName": "swrc:firstName",
  "uri": "http://swrc.ontoware.org/ontology#firstName",
  "type": "property",
  "score": 0.32398805
},
```


Figure 10 The first three results of the search for the term "firstName" in LOV.

Table 2 The results of constructing possible combinations of a complex term

Word	Possible Combinations
first name	first_name, first-name,firstName, firstname
FirstName	first_name, first-name,first name, firstname
first_name	firstName, first-name,first name, firstname
...	...

of text. Their approach offers a significant increase in accuracy and recall compared to alternative solutions.

TextRazor's relationship extraction system can be used to construct RDF triples since an RDF triple consists of (subject-predicate-object) where the entities represent the subject and the object, while the relationship between them represents the predicate.

4.3 Vocabularies Selection

Another critical step is the selection of the vocabularies to be used in describing data.

A user who knows his dataset may wish to describe it in a specific vocabulary such as FOAF, SKOS... or use any vocabulary.

In this step, we define, among the list of vocabularies retrieved from the LOV API, the vocabularies we want to use in the conversion (all vocabularies are selected by default).

The results of the mapping step will strongly depend on the choice made in this step: for example, if we choose to describe the data in the FOAF vocabulary and we look for the term first name, we will have foaf:firstName as RDF property, but if we choose the FOAF and SIOC[5] vocabularies we will have the two properties *foaf:firstName* and *SIOC:first_name.*

4.4 Data Mapping

In this phase, we aim to describe the data extracted from the previous phases in the selected vocabularies:

4.4.1 Tabular data

We take an example of a simple CSV input file:

first_name,last_name,age
Bob, Smith,25
David, Johnson,72

We have to:

1. Choose or create a class representing each row of the table: in the example, we can notice that each row represents a person, and therefore, the appropriate class is the person class. In this case, we have two choices:
   a. Use the person class from an existing vocabulary retrieved from the LOV API (e.g., foaf: person).
      Or
   b. Create a new class with the attributes:
      i. label: for the name of the class.
      ii. subClassOf: if it inherits from another parent class.
      iii. Comment: for the description of the class.
2. Choose headers that can take part in each row identifier. This identifier will be used in the construction of the subject URI: in the previous example, if we define the columns first_name and last_name as identifiers, the subject URI of the first row will be 'http://www.example.com/Bob_Smith.' If no identifier is specified, the index of the row will be considered as an identifier, and so for the same example, we will have: 'http://www.example.com/1' as subject URI.

[5] http://rdfs.org/sioc/spec/

Table 3 The result of CSV file mapping

rowClass	foaf: person	
rowId	first_name, last_name	
headers	first_name	foaf:firstName
	last_name	foaf:lastName
	age	foaf:age

3. For each header: search for possible URI terms in LOV for each possible combination of the term (after preprocessing) sorted in descending order of score. This score is a metric calculated to give a higher value for most used terms and detailed [18].

If we take the header first_name, we will search for the terms corresponding to all possible combinations of the word (first_name, first-name, firstName, first name) and sort the list in descending order of score. In this case, the most appropriate term will be foaf:firstName.

If none of the terms are suitable for our property, a new property can be created.

The result of this step for the previous example file is shown in Table 3.

4.4.2 Text data

We take the example of a simple sentence:

Bill Gates is the founder of Microsoft Corporation

The result of the previous step should be:

- Subject: **Bill Gates**
- Predicate: **is the founder**
- Object: **Microsoft Corporation**

We notice that all terms are in string format, and therefore, the linked data principles are not respected.

During this step:

a. The predicate of the triple is mapped to an RDF property, and therefore, we are looking for the LOV terms corresponding to the predicate. In the example, the predicate 'is the founder' can be mapped to an RDF property from an existing vocabulary (e.g. "DBpedia-owl: founder") or a new property can be created.

Table 4 The result of the text file mapping

Bill Gates	http://dbpedia.org/resource/Bill_Gates
is the founder	DBpedia-owl: founder
Microsoft Corporation	http://dbpedia.org/resource/Microsoft

Table 5 The result of the CSV file conversion

Subject	Predicate	Object
www.example.com/Bob_Smith	A	foaf: person
www.example.com/Bob_Smith	foaf:firstName	"Bob"
www.example.com/Bob_Smith	foaf:lastName	"Smith"
www.example.com/Bob_Smith	foaf:age	"25"
www.example.com/David_Johnson	A	foaf:person
www.example.com/David_Johnson	foaf:firstName	"David"
www.example.com/David_Johnson	foaf:lastName	"Johnson"
www.example.com/David_Johnson	foaf:age	"72"

b. The subject and object of the triple can be searched using the DBpedia API to get the URI of each of them.
c. The result of this step for the previous text file is represented in Table 4.

4.5 Data Conversion

This step is dedicated to transforming the extracted data into linked data represented with a set of triples constructing an RDF graph.

During this step:

4.5.1 Tabular data

a. The row identifier is mapped to the triple subject.
b. The header is mapped to the triple predicate.
c. The cell value is mapped to a triple object.

If we take the example of the previous CSV file again and with the mapping represented in Table 3, we will have the list of triplets represented in Table 5 with the turtle format.

4.5.2 Text data

a. The sentence subject is mapped to the triple subject.
b. The sentence object is mapped to the triple object

Table 6 The result of the conversion of the text file

Subject	Predicate	Object
http://www.example.com/Bill_Gates	rdfs:seeAlso	http://dbpedia.org/resource/Bill_Gates
http://www.example.com/Microsoft	rdfs:seeAlso	http://dbpedia.org/resource/Microsoft
http://www.example/Bill_Gates	dbpedia-owl:founder	http://www.example.com/Microsoft

c. The relation between the subject and the object (represented by the verb) is mapped to the triple predicate.

The subject and predicate are always in URIs constructed at the data mapping stage while the object can be a URI or a literal.

The sentence conversion of the previous text file with the mapping defined in Table 4 is represented in Table 6.

This step ends with the serialization of the RDF graph produced using one of the RDF serialization formats (XML, JSON-LD, Turtle…) to have a dataset described in RDF ready for download.

At the end, the RDF file resulting from the conversion of the previous csv file into turtle format, for example, should be as represented in the following listing:

```
@prefix dataset: <http://www.example.com/>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
dataset:Bob_Smith a foaf:Person;
foaf:age 25;
foaf:firstName “Bob”^^xsd:string;
foaf:lastName “Smith”^^xsd:string.
dataset:David_Johnson a foaf:Person;
foaf:age 72;
foaf:firstName “David”^^xsd:string;
foaf:lastName “Johnson”^^xsd:string.
```

4.6 Data Documentation

This step is optional, and it aims to provide additional information about the converted dataset: the metadata that can provide information about the dataset itself, such as the dataset owner, the license, the dataset description, date of the last update.

Table 7 Terms from the Dublin Core vocabulary used to express output dataset metadata

Term	Description
dc: title	The dataset title
dc: creator	The dataset creator
dc: subject	The subject in which the dataset is inscribed
dc: description	The dataset description
dc: date	The dataset creation date
dc: license	A legal document giving official authorization to use the dataset

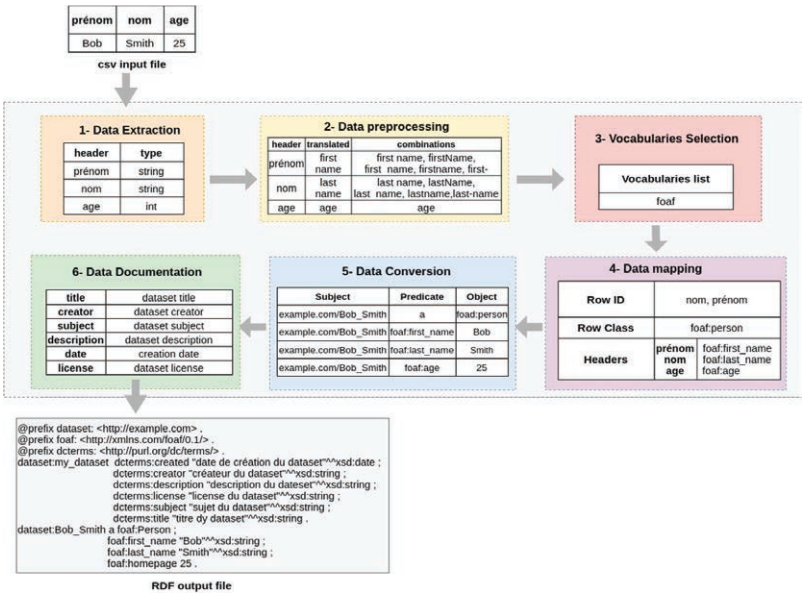


Figure 11 The process of transforming a CSV file into linked data in turtle RDF format.

This information is necessary because anyone can publish anything on the Web and verify the published data; we often refer to these metadata.

To describe the dataset's metadata as linked data, we have used the "Dublin Core" vocabulary, which offers several terms to describe the metadata. Table 7 represents the terms we have chosen to describe the metadata of the resulting dataset.

Figures 11 and 12 represent, respectively, the processes of transforming a CSV file and a Text file and the operations performed in each step to transform the input file into an RDF output graph.

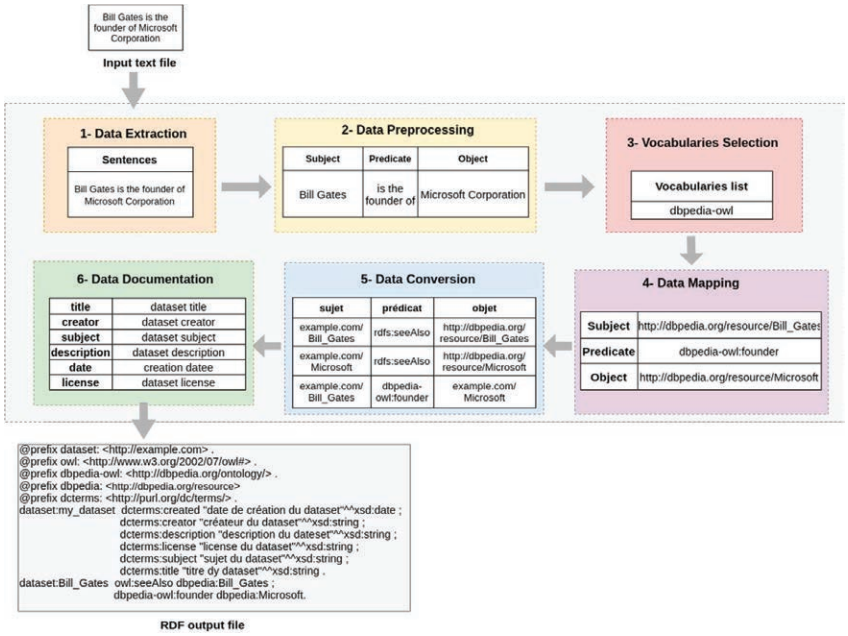


Figure 12 The process of transforming a text file into linked data in RDF Turtle format.

5 Implementation and Tests

5.1 Implementation

In order to guarantee a modular architecture, which facilitates the evolution and the reuse of the tool, and a separation between data, processing, and presentation, we have divided the tool into four layers represented in the Figure 13:

- **Client layer:** to present data to the user, the only layer with which the user interacts.
- **Processing layer:** to retrieve data from the client layer, perform processing and return the results.
- **The service layer:** represents all the external APIs used in the different steps (LOV API, TextRazor API, DBpedia API).
- **Persistence layer:** for everything concerning the representation and storage of data.

The interactions between the different layers occur in the application via the MVC (Model-View-Controller) design pattern, which defines our tool's software architecture.

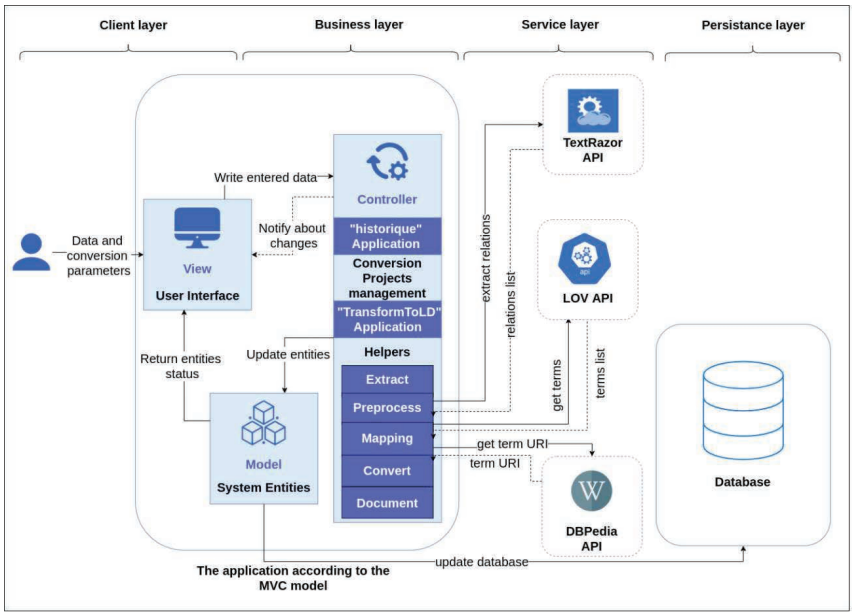


Figure 13 The implemented tool architecture.

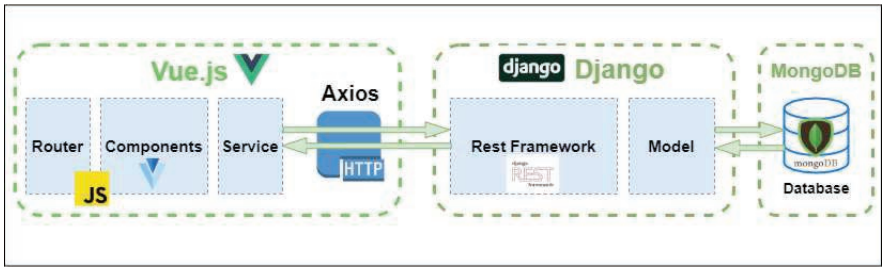


Figure 14 Technologies used to implement the proposed approach.

We have decided to implement our approach as a web application, and this is due to the following reasons:

- No installation for users
- Ease of updates and upgrades
- A better user experience
- Capability to scale the application.
- Security

Figure 14 represents the technologies used to implement the solution:

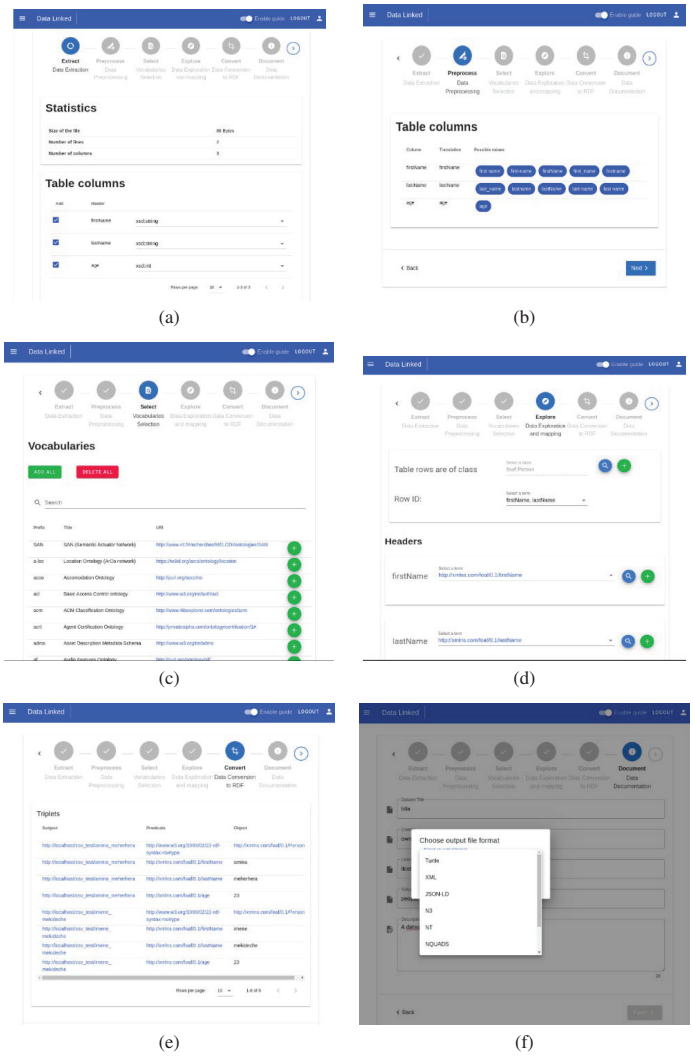


Figure 15 (a) Extraction phase. (b) Preprocessing phase. (c) Vocabularies selection phase. (d) Mapping phase. (e) Conversion phase. (f) Documentation and choice of serialization format.

The source code is available at https://github.com/MinaMeh/TransformToLD and the figures above show the developed tool where we see in (Figure 15(a)) that data is extracted; for tables, it is about extracting row and columns (with their types), then these columns are preprocessed

(Figure 15(b)) in order to prepare them before mapping to LOV terms (Figure 15(d)) using selected vocabularies (Figure 15(c)). Finally, a list of RDF triples is generated (Figure 15(e)), and the converted dataset is documented using Dublin Core vocabulary before choosing one of the serialization formats and download it (Figure 15(f)).

The difference with text files is that we extract row and columns for tabular files, while for text files, we extract sentences and their structures. Otherwise, for files containing texts and tables, we extract both of them.

5.2 Tests

In this section, we will perform and analyze our application tests to verify the quality of the tool and ensure that it meets the needs.For this purpose, we have chosen two types of tests to perform:

a. Functional tests: To verify that the tool produces good quality linked data and provides the required functionalities.
b. Performance tests: They aim to verify that the tool remains functional for quite large input file sizes within an acceptable time.

The characteristics of the hardware resources of the machine used to perform the tests are represented in Table 8:

Table 8 The characteristics of the hardware resources used for testing

RAM	8GB
CPU	Intel§Core^{TM} i5-4200U CPU @ 1.60GHz $\times$ 4
GPU	Intel§Haswell Mobile
Operating system	Debian GNU/Linux 10 (buster) -64 bits
Disk Space	474.4 GB
Internet speed	2MO

5.2.1 Functional tests

This section verifies that the tool transforms the input file (CSV, Text, HTML, PDF, Image) into linked data in RDF format with different serialization formats (Turtle, XML, JSON-LD. . .).

To do this:

– We have downloaded datasets from the Web with different formats (CSV, text, HTML, pdf, image).
– We converted the datasets using the implemented tool.

Figure 16 The syntax validation result of the RDF file output with Validata.

Table 9 Verification of the four principles of linked data for the result file

Principle 1	Verified	URIs are used to identify objects
Principle 2	Verified	Objects are dereferenceable.
Principle 3	Verified	Data are represented in RDF
Principle 4	Verified	Linking to other datasets is ensured by reusing existing vocabularies and also by using rdfs:seeAlso

– We checked the linked data in output using two criteria:

a. Syntax: to make sure the data is described using a valid RDF representation, we have used the online tool Validata,[6] and the results are represented in Figure 16.

b. Quality: by verifying that the output RDF graph respected the four linked data principles. The results are represented in Table 9.

5.2.2 Performance tests

We have performed performance tests to measure the scalability of our tool, and it is about measuring the execution time of the transformation for different file sizes.

The metrics used in our tests are represented in Table 10.

- **CSV files tests:**

The table and figures below represent the results of CSV file tests.
We can notice that:

1. The total execution time does not only depend on the input file size (Figure 17(c)), but it also depends on the number of columns (Figure 17(a)): this can be explained by the fact that the most expensive step is usually the mapping step, which deals with the columns by trying to map each column to a term of the LOV.

[6]https://www.w3.org/2015/03/ShExValidata/

Table 10 The metrics used to evaluate the tool performance

Metrics	CSV Files Tests	Text Files tests	Complex Files Tests
		source file size	
		total execution time	
		extraction time	Tables number
		preprocessing time	
		mapping time	
		documentation time	
			Paragraphs number
		triples number	
	Rows number	Sentences number	
	Columns number		

Table 11 CSV file transformation performance test results

File size (kb)		1,5	15,1	131,5	1800	6700	8000
Number of rows		19	131	2129	5105	211302	36512
Number of columns		6	17	3	41	4	18
Step	**Extraction (s)**	0,90	0,01	0,01	0,06	0,08	0,19
	Preprocessing(s)	4,00	9,44	3,61	27,06	2,19	10,30
	Alignment(s)	4,68	32,09	5,36	73,52	5,25	22,93
	Conversion(s)	0,22	0,09	0,43	6,93	44,79	20,60
Total (s)		9,79	41,63	9,41	107,57	52,31	54,02
Number of triples		114	2227	6387	209305	845208	657216

2. The preprocessing and mapping steps' execution time depends on the number of the columns (Figure 17(a)), while the conversion step depends mainly on the rows number (Figure 17(b)). On the other hand, the extraction step is the least expensive (Figures 17(a)–17(c)).
3. We distinguish two cases (Figures 17(a) and 17(b)):
   a. If the row number is huge, the conversion step is the most expensive.
   b. If the number of columns is huge, the mapping step is the most expensive.

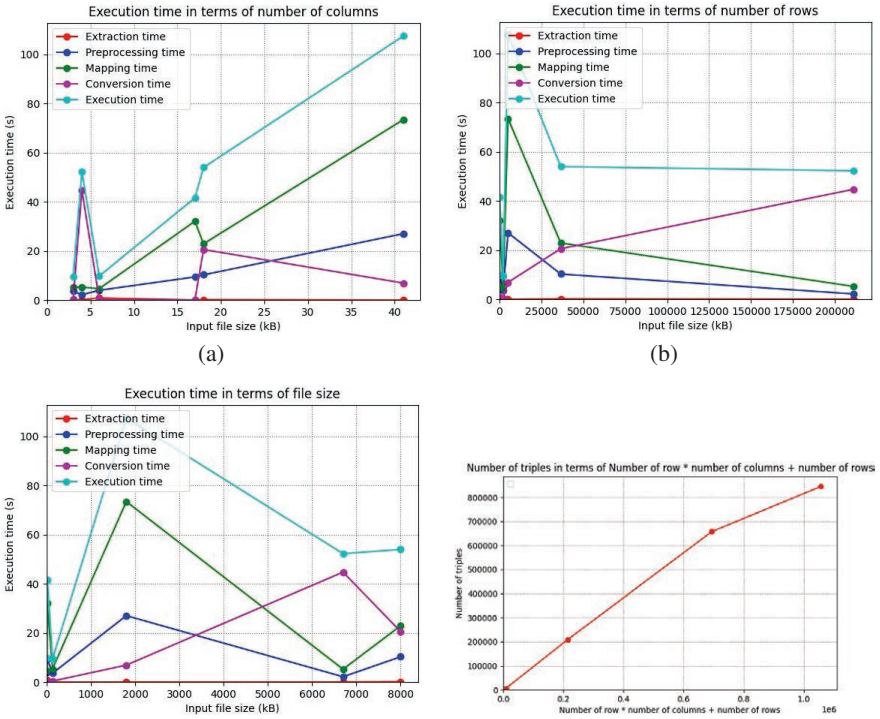


Figure 17 (a) Execution time in terms of number of columns. (b) Execution time in terms of number of rows. (c) Execution time in terms of file size. (d) Number of triplets in terms of number of rows + number of rows*number of columns.

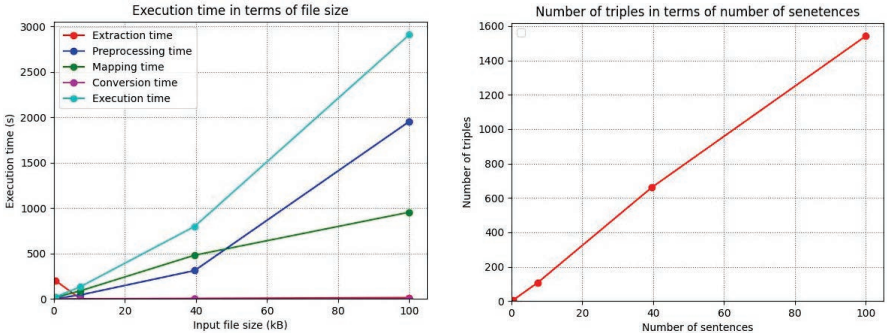
4. The number of triples = number of rows * number of columns + number of rows (Figure 17(d)): this can be explained by the way the conversion is done:
    i. For each line:
        1. We create the triple of the row class (with the RDF isA property).
        2. For each column:
            a. We create an RDF triple for the property mapped to the column name.

- **Text Files tests:**

The table and the figure below represent the results of text files tests.

Table 12 Text file transformation performance test results

File size (kb)		1,5	7,4	39,6	100
Number of sentences		4	57	387	1412
Step	**Extraction (s)**	0,84	1,93	5,11	12,67
	Preprocessing(s)	3,77	42,78	311,71	1953,23
	Alignment(s)	17,10	88,14	480,92	954,3
	Conversion(s)	0,01	0,02	0,01	1,2
Total (s)		21,71	132,85	797,65	2908,59
Number of triples		6	107	661	1542



(a) (b)

Figure 18 (a) Execution time in terms of file size. (b). Number of triples in terms of number of sentences.

Table 13 The results of the performance tests of the transformation of HTML files

File size (kb)	1,4	24	162	3100
Number of tables	2	8	13	540
Number of paragraphs	3	7	10	286
Extraction execution time (s)	2,5	5,64	16,7	421,97

We can notice that:

1. The execution time of the extraction and conversion steps is negligible (Figure 18(a)): this can be explained by the fact that during extraction, we only extract the sentences, and during conversion, we generate the triplets from the terms resulting from the mapping step.
2. The preprocessing step is the most expensive (Figure 18(a)), and it increases exponentially depending on the file size; this comes back to

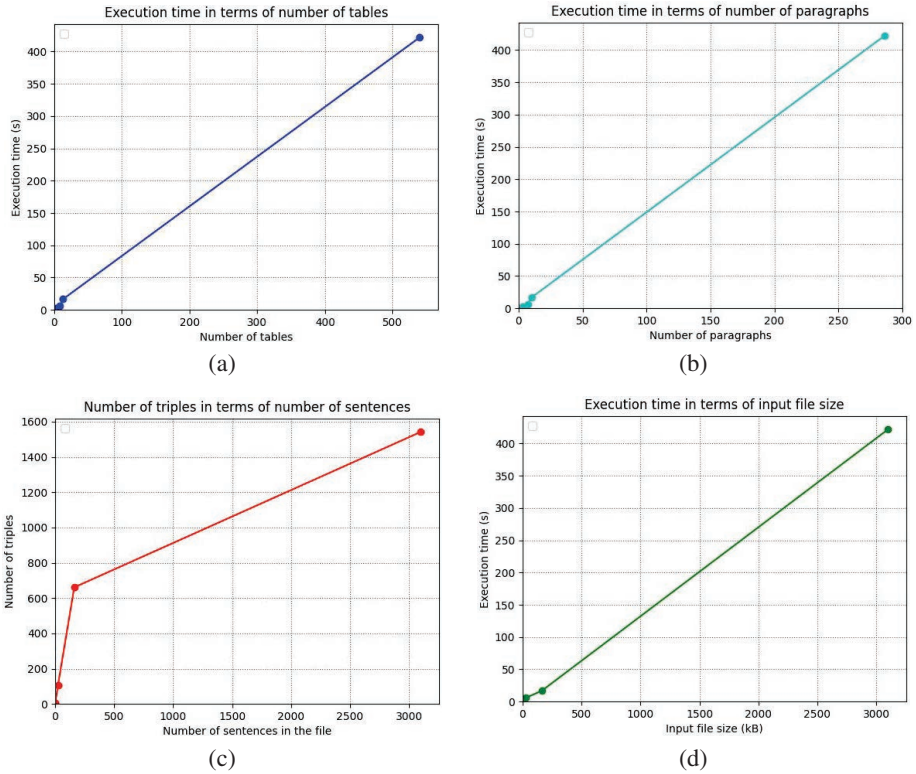


Figure 19 (a) Extraction time in terms of number of tables. (b) Extraction time in terms of number of paragraphs. (c) Number of triples in terms of number of sentences. (d) Number of triples in terms of file Figure size.

the need to use NLP techniques to extract entities and the relationships between them.

3. The mapping step is moderately expensive (Figure 18(a)) and depends on the number of sentences since we map each sentence's relationship to a LOV term.
4. The number of triples is higher than the number of sentences since a sentence can contain more than one relationship (Figure 18(b)).

- **HTML files tests:**

We have measured only the execution time of the extraction step for complex files since, after these steps, the files will be converted into tabular and text files.

The table figures below represent the results of the performance tests in the case of HTML files.

We notice that:

1. The extraction time increases linearly with the number of paragraphs and tables (Figure 19(a) and 19(b)).
2. Paragraph extraction time is slightly longer than tables' extraction since we extract sentences in addition to paragraphs (Figures 19(a) and 19(b)).
3. The number of triplets depends strongly on the number of sentences in the file (Figure 19(c)).
4. The number of triples is linear with the input file size (Figure 19(d)).
5. When the file contains both texts and tables, the extraction time is longer.

6 Conclusion

In the paper, we presented a new approach to transform open web data to linked data that comes with solutions to many of the problems of existing approaches in the literature: Our approach is generic and can be used in any domain to convert any dataset. It also accepts five file formats, which are the most used for the publication of data on the Web (CSV, text, HTML, pdf, image), and it can be extensible for other file formats easily. Besides, data resulting from the conversion is strongly linked to external datasets using existing vocabularies.

Besides, our approach can be used by semantic web experts and non-experts since it offers an automatic mode where default parameters are taken into consideration and a semi-automatic mode where the user can change the conversion parameters.

We have used a modular architecture to ease the reuse of different implemented modules and components in terms of implementation. Finally, the implemented approach has been tested and approved with both functional and performance tests, which revealed satisfactory results and confirmed our approach's efficacy.

For future considerations, the auto-detection of row class in CSV files according to attributes can be studied using SPARQL queries on vocabularies. Considering tables with complex structures may also be considered in future works. Finally, an effort can be made to design a new model with better accuracy to extract relations in texts to enhance output linked data quality.

References

[1] Guerrini, M. and T. Possemato, *Linked data: a new alphabet for the semantic web.* JLIS. it, 2013. **4**(1): p. 67.

[2] Wood, D., et al., *Linked Data* 2014: Manning Publications Co.

[3] Berners-Lee, T., J. Hendler, and O. Lassila, *The semantic web.* Scientific American, 2001. **284**(5): p. 34–43.

[4] Bizer, C., R. Cyganiak, and T. Heath, *How to publish linked data on the web.* 2007.

[5] Heath, T. and C. Bizer, *Linked data: Evolving the web into a global data space.* Synthesis lectures on the semantic web: theory and technology, 2011. **1**(1): p. 1–136.

[6] Hitzler, P., M. Krotzsch, and S. Rudolph, *Foundations of semantic web technologies* 2009: CRC press.

[7] Patel, A. and S. Jain, *Present and future of semantic web technologies: a research statement.* International Journal of Computers and Applications, 2019: p. 1–10.

[8] Miller, R.J., *Open data integration.* Proceedings of the VLDB Endowment, 2018. **11**(12): p. 2130–2139.

[9] Bizer, C., et al. *Linked data on the web (LDOW2008)*. in *Proceedings of the 17th international conference on World Wide Web*. 2008.

[10] Zemmouchi-Ghomari, L., *Linked Data: A Manner to Realize the Web of Data*, in *Handbook of Research on Technology Integration in the Global World* 2019, IGI Global. p. 87–113.

[11] Sharma, K., U. Marjit, and U. Biswas, *Automatically converting tabular data to rdf: An ontological approach.* Int J Web Semant Technol, 2015.

[12] Lebo, T. and G.T. Williams. *Converting governmental datasets into linked data*. in *Proceedings of the 6th International Conference on Semantic Systems*. 2010.

[13] Hallo, M., S. Luján-Mora, and J.C. Trujillo Mondéjar, *Transforming library catalogs into linked data.* 2014.

[14] Mulwad, V., T. Finin, and A. Joshi, *Automatically generating government linked data from tables.* UMBC Faculty Collection, 2011.

[15] Jaglan, G. and S.K. Malik. *LOD: Linking and Querying Shared Data on Web*. in *2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. 2018. IEEE.

[16] Atemezing, G., et al., *Transforming meteorological data into linked data.* Semantic Web, 2013. **4**(3): p. 285–290.

[17] Gupta, A., et al. *Integrating linked open data with unstructured text for intelligence gathering tasks*. in *Proceedings of the 8th International Workshop on Information Integration on the Web: in conjunction with WWW 2011*. 2011.
[18] Vandenbussche, P.-Y., et al., *Lov: a gateway to reusable semantic vocabularies on the web.* Semantic Web Journal, 2015.

Biographies

Amina Meherhera, currently Data Engineer at Dataimpact since Septemeber, 2020. She attended the Ecole Nationale Supérieure d'Informatique of Algiers (ESI-ex INI) from 2015 to 2020 and she graduated on 2020 with master degree and engineering degree in computer science. Her master graduation project was about state of the art of existing approaches to transform web open data to linked data and her engineering project was about proposing a new approach that covers the limits of the approaches studied in the state of tha art. The work was a part of a governmental project held by LMCS (Laboratoire des Méthodes de Conception des Systèmes) where she worked as a researcher from September, 2019 to Sepetember, 2020.

Imene Mekideche is a Junior Software engineer in the National Employment Agency, Algeria with a passion for computer science, web development and technology. She earned her engineer's and Master's degrees in Computer Science Engineering from Ecole Nationale Supérieure d'Informatique, ESI. The graduation's project was an opportunity to be involved in the scientific research field, the topic was about proposing a new approach for transforming open data to linked data. She is currently, a freelancer frontend developer with Unchained Agency (United Arab Emirates), in parallel of freelance, her main job is with the National Employment Agency – Algeria as a software engineer.

Leila Zemmouchi-Ghomari, received her PhD in Computer Science from Ecole Nationale Supérieure d'Informatique (ESI), Algiers, in January 2014. Her research interests focus on ontology engineering, knowledge engineering, semantic web, linked data and open data. She works currently as an Associate Professor at ENST: Ecole Nationale Supérieure de Technologie, Algiers, Algeria.

Abdessamed Réda Ghomari, received his PhD in Computer Science (Information System) from ESI (Ecole Nationale Supérieure d'Informatique), Algiers, 2008. He is currently a Full Professor at ESI, Algiers, Algeria. Since 2001, he is an Information System Management Team Head at the LMCS (the Laboratory of Systems Design Methodologies). His research interests focus on Risks Management, Data governance, Open Government data, Collective intelligence and Crisis Informatics.