Re-engineering Web Application towards Linked Data: a Model-Based Approach

Benamar Bouougada[1], Djelloul Bouchiha[2], Yassine Ouhammou[3] and Mimoun Malki[4]
[1]EEDIS Laboratory, University of Djillali Liabès, Sidi BEL-ABBES, Algeria
[2]EEDIS Lab., Ctr Univ Naama, Inst. Sciences and Technologies, Dept. Mathematics and Computer Science, Naama, Algeria
[3]École Nationale Supérieure de Mécanique et d'Aérotechnique (ENSMA), Futuroscope, France
[4]LabRI-SBA Lab., École Supérieure en Informatique, Sidi Bel Abbes, Algeria

Abstract *Nowadays, Wide World Web is a wide network of information resources found in documents, such as HTML pages, PHP, etc. Most are intended for consumer uses, whether by human or machines (i.e., programs). So the Web grows with the emergence of new technologies, such as Web services, mobile applications and Web applications. These technologies manipulate data in multiple formats in a hidden or unusable way to users. In parallel, there is also another growth of desire for directly accessing data on the Web. However, the current Web cannot meet this need. So, we have for instance to switch to the semantic Web by reengineering our classic Web applications into RDF Linked data. This can be justified by the fact that (in the semantic Web) data are represented in RDF format which makes them directly available to users. In this paper, we propose a model-based approach to transform Web applications into semantic ones. This is done by extracting data from the Web applications, and transforming them into RDF format. These data are extracted from HTML files, in particularly from tables, lists and links. This engineering process is capitalized thanks to the use of model driven engineering settings. It takes as input a set of elements: (1) the input file which contains the data that we want to transform; these data are extracted from the Web application, (2) two meta-models, namely, the source meta-model (for HTML) and a target meta-model (for RDF), and (3) the transformation rules. Executing this process generates an output file in RDF format, which is one of the most important technologies of the Semantic Web, also known as Web of Data.*

Keywords: *RDF, Model-driven engineering, Re-engineering, Web application, Semantics, Linked data.*

Received December 22, 2017; Accepted March 23 2018

1. Introduction

Nowadays, the Web is a collection of interconnected documents which contain an amount of data. Often, we find data in some elements of HTML pages like tables and lists. These structures were particularly chosen because there are billions of tables storing data on the Web [6]. Traditionally, these data are automatically generated from the database.

Since databases are located in servers, it is not possible to be accessed directly through the Web. Hence, the only possible access to the data is via the HTML pages. However, because of their format, it is very hard to process HTML data. Therefore, they require to be transformed to another format, notably RDF Linked data [4], in order to be more exploitable and machine-interpretable. Practically, we apply a reengineering process to transform a Web application into a Semantic Web application. Consequently, we allow publishing RDF data on the Web.

Various research challenges have been addressed by the semantic Web community, such as ontology alignment [1], maintaining RDF links [2] and re-engineering Web applications [15, 19, 10, 23, and 16]. In this paper, we are working on re-engineering Web applications towards semantic Web. Web applications consist of a set of HTML pages and PHP scripts. We focus on transforming HTML pages since they are rich in data.

Here, we propose a model-based approach that allows transforming Web applications into RDF Linked Data. It receives as input client-side HTML pages, and produces as output RDF triples. The proposed approach consists of three steps: (1) Preprocessing step, which prepares HTML inputs to the next step, (2) Transformation step, which is a reengineering process according to the MDE principles, and (3) Refinement, which consists in refining results to have the final RDF outputs. This approach is implemented by a tool called HTML2RDF.

The remainder of the paper is organized as follows. In the next section, we introduce some important notions to help to understand the issue addressed in this paper. In Section III, we present a running example to illustrate our work. Section IV describes the proposed approach. Section V sketches the system implementations. It also presents and discusses the obtained result by our HTML2RDF tool on some Wikipedia pages. In Section VI, we discuss some related work. Finally, Section VII concludes the paper and presents some perspectives.

2. Background

In order to make this research self-contained and straightforward, this section gives some required concepts to understand our approach, notably RDF, MDE and Web applications.

2.1. Resource Description Framework (RDF)

As shown in Figure 1, the Semantic Web stack consists of several layers; each one represents a language [11].

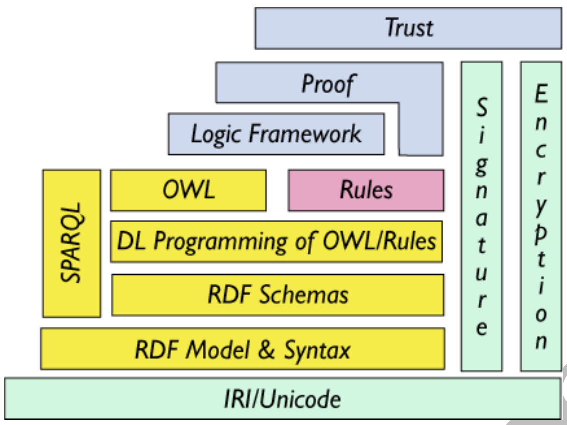


Figure 1. Web semantic layers.

We see that RDF constitutes the base layer for the semantic Web. It is used to represent semantically the data existing on the Web. It also describes the meta-data. RDF uses a graphical format to describe data and relationships between entities. Each model or language, using RDF, is necessarily a part of the Semantic Web.

The main idea of RDF is that every data on the Web must have a unique identifier called URI (Uniform Resource Identifier), and each one can be connected to another data item. So, RDF makes URI relationships between data in the Web. The Semantic Web developers create data and connect them with URIs. In this way, a set of data will be distributed on the Web. RDF presents the information as a set of resources where each one is composing a set of statements or triples. Each statement must be written in the following format, and in the order: subject-predicate-object [12]:

- Subject: is an IRI or a blank node (any things in the world)
- Predicate: is an IRI (name of relation between subject and object)
- Object: is an IRI, a literal or a blank node (may be also a subject or simply value of a subject).

2.2. Model Driven Engineering (MDE)

The model-driven engineering is a generative process enabling to get, treat and produce information from other data derived from structured entities called models. MDE settings are mainly based on modeling, meta-modeling and transformation aspects. Another important MDE aspect is the conformity relationship. A model conforms to the meta-model as a program conforms to the programming language's grammar in which it is written. This means that a meta-model describes the different types of model elements and how they are arranged.

Therefore, the MDE is considering models and software as first class entities. The model represents any aspect of software, and it consists of a set of model elements.

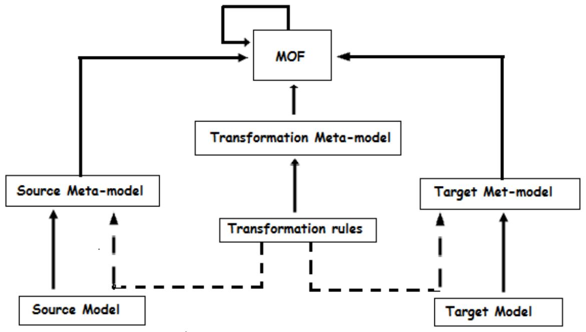


Figure 2. A model-based transformation process according to MDE.

Figure 2 shows a model-based transformation process. It allows to transform any source model *Ma* (in our case HTML elements) that conforms to *MMa* (source meta-model of Figure 9), into a target model *Mb* (RDF triples) that conforms to *MMb (target* meta-model of Figure 11). The transformation is driven by a set of rules expressed as transformation model (Listing 1, 2, 3, 4), which is itself conforms to a transformation meta-model [5].

2.3. Web application

As shown in Figure 3, Web application consists of a set of pages divided into two subsets: the first one contains Server-side pages, such as PHP, JSP and HTML pages. The second subset is Client-side pages, which are only HTML pages built dynamically by the Web server.

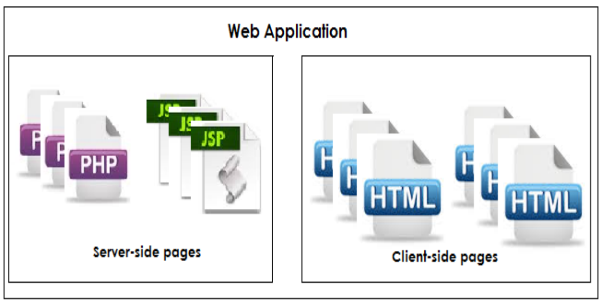


Figure 3. Web application components.

Often, in the World Wide Web, we find a lot of Web applications, but we haven't access to their database. We can only access the client pages. In our system, we consider these client pages, and we aim to transform them into semantic data to achieve our goal.

3. Running Example

In this section, we introduce a running example showing how to transform a Web application into semantic one, in particular, mapping HTML documents to RDF triplets.
As shown in Figure 4, for each HTML document, an RDF file is generated. The triples are then constructed from HTML data elements, such as lists, tables, etc.

HTML
Transformation process
RDF

Figure 4. Transforming HTML documents into RDF files.

The following figure (Figure 5) shows the transformation of an HTML list into RDF. The side (a) of the figure represents a browser preview containing a data list. The side (b) shows the RDF triple corresponding to side (a) that we aim to obtain automatically through a capitalized transformation process.

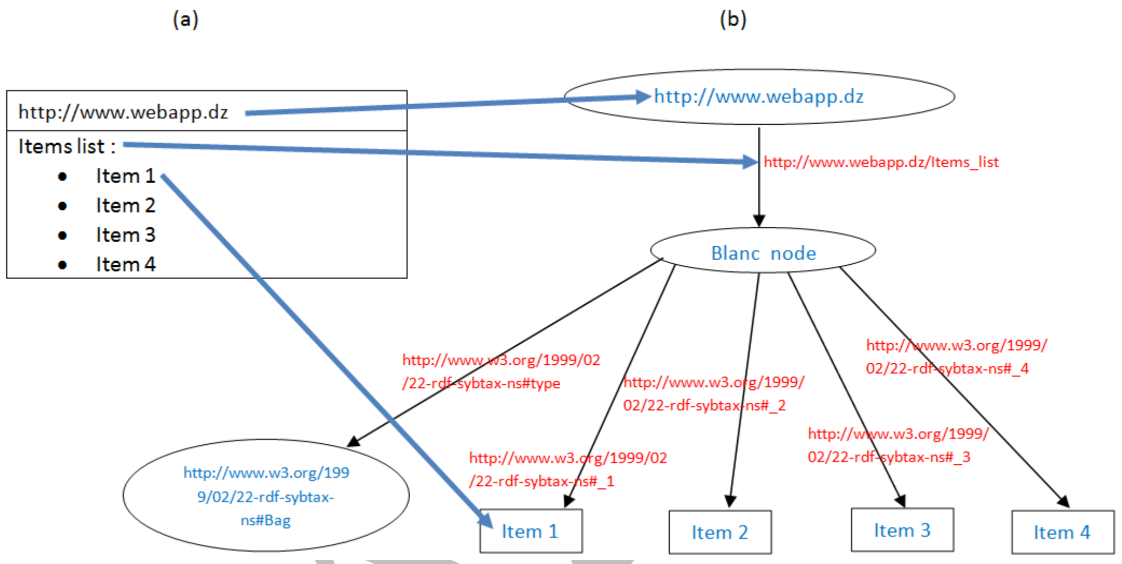

(a)
(b)
http://www.webapp.dz
Items list :
Item 1
Item 2
Item 3
Item 4
http://www.webapp.dz
http://www.webapp.dz/Items_list
Blanc node
http://www.w3.org/1999/02/22-rdf-sybtax-ns#type
http://www.w3.org/1999/02/22-rdf-sybtax-ns#_1
http://www.w3.org/1999/02/22-rdf-sybtax-ns#_2
http://www.w3.org/1999/02/22-rdf-sybtax-ns#_3
http://www.w3.org/1999/02/22-rdf-sybtax-ns#_4
http://www.w3.org/1999/02/22-rdf-sybtax-ns#Bag
Item 1
Item 2
Item 3
Item 4

Figure 5. Transformation example.

We note in this example that HTML elements, notably lists, represent models that carry data. These data will be represented in RDF format. It is clear that whenever a transformation of HTML data into RDF is desired, the same transformation rules are applied whatever the data to be transformed, either tables, lists or links.

So we have as input models (lists, tables and, links) that will undergo a transformation process to get an output model (RDF). This idea is similar to the MDE paradigm. It is based on the definition of the source and target meta-models, and a set of transformation rules. In our case, we must generalize our models (input, output) to construct two meta-models: a source meta-model (HTML input model) and a target meta-model (RDF output model).

In the remainder of the paper, these meta-models and these mapping rules are detailed, and they are all based on model-driven engineering (MDE) settings.

4. Proposed Approach

We propose a process that consists of three steps as illustrated in Figure 6. The first one is *pre-processing* (or Data extraction) where candidate data are extracted and represented in an appropriate format. The second step constitutes the kernel of the system which is the *model based transformation process*. The last one is the *refinement* of result obtained by the previous step. In the sequel, we detail the role and the content of each step.

4.1. Preprocessing Step

In this step, our system takes client pages as input. Preprocessing consists of two sub-steps: extraction and adaptation. The preprocessing or data extraction aims at preparing data for the next step.
We can summarize this task in three words: extract and adapt to build!!! So we extract what? We adapt to build what? The response to these questions is easy as far as you think.

- Extracting means that we only take HTML elements that contain data, like <table>.... </table>, <ol>... </ol>, <ul>... </ul> and <dd>... </dd>, because these elements contain data which comes often from hidden database. The others elements like <img>,

<frame>, etc. are ignored and removed from the HTML client pages.

- Adapt to Build means that the result of the previous stage is adjusted to be conform to the source Meta-model defined in the next step, eg. <ol>.... </ol> is transformed into <list>...</list>.

Now, we take the example presented in the previous section, and we explain how the data extraction and adaptation step is done. Figure 7 displays the preprocessing step.

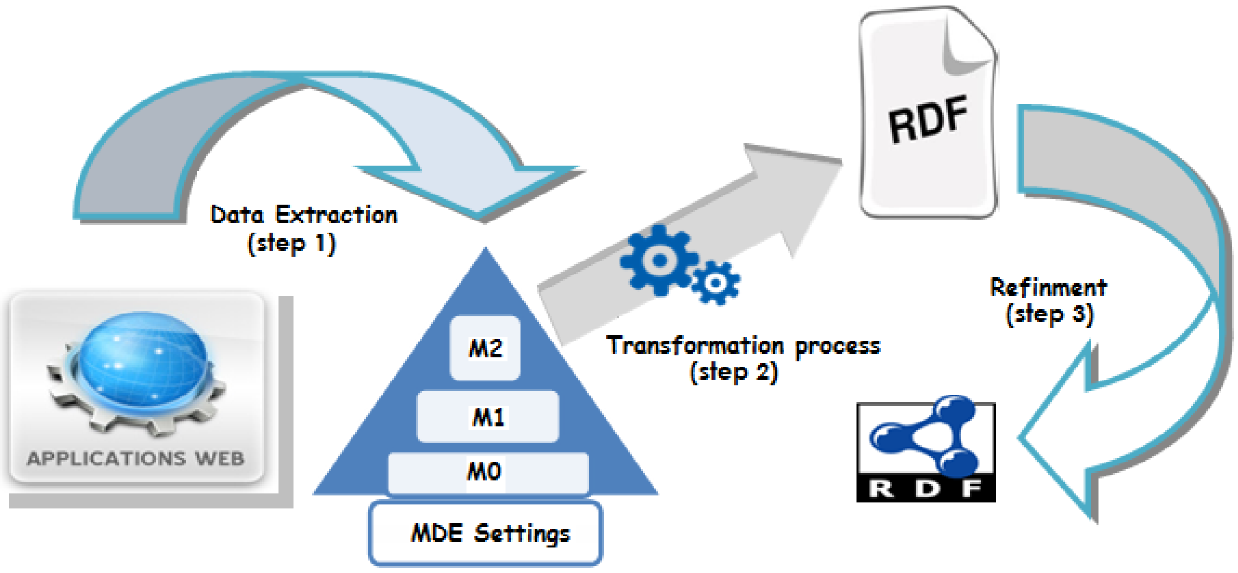
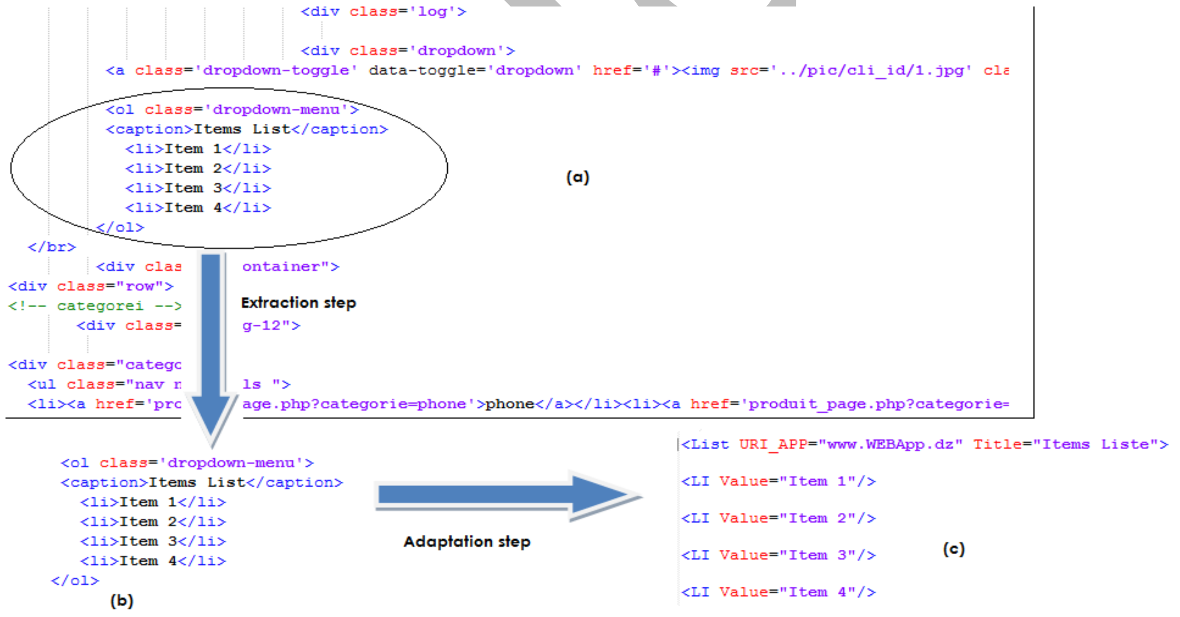


Figure 6. Architecture of our system.



```
                    <div class='log'>

                    <div class='dropdown'>
  <a class='dropdown-toggle' data-toggle='dropdown' href='#'><img src='../pic/cli_id/1.jpg' cla

  <ol class='dropdown-menu'>
  <caption>Items List</caption>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
    <li>Item 4</li>
  </ol>
</br>
    <div clas       ontainer">
<div class="row">
<!-- categorei -->
   <div class=      g-12">

<div class="categc
  <ul class="nav r     ls ">
  <li><a href='prc     age.php?categorie=phone'>phone</a></li><li><a href='produit_page.php?categorie=
```

```
  <ol class='dropdown-menu'>
  <caption>Items List</caption>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
    <li>Item 4</li>
</ol>
```

```
<List URI_APP="www.WEBApp.dz" Title="Items Liste">

<LI Value="Item 1"/>

<LI Value="Item 2"/>

<LI Value="Item 3"/>

<LI Value="Item 4"/>
```

Figure 7. Preprocessing task: extraction and adaptation of data.

As shown in Figure 7, the upper side (a) shows the HTML code of the running example, from which we apply our pre-processing process, namely the extraction and adaptation of the data. The result is shown in the lower side of Figure 7, i.e. (b) and (c) respectively. The language (tags) used for the adaptation is defined by the source meta-model, which will be detailed in the following section. The result of the preprocessing step will be automatically transmitted to the next step.

4.2. Reengineering Step based on MDE setting

This second step constitutes the kernel of our system. It is based on the model-driven engineering (MDE), and it is composed of three layers as defined in MDE settings (Figure 8). Layer M0 contains the input file received from the previous task (pre-processing) and the output file (RDF file) that we want to create with this system. The layer M1 includes three files, especially two meta-models (MM), one for defining the source MM and the other for the target MM; the third file is the transformation engine. Finally, the last layer consists of the meta-meta-model of all files defined in the previous layer. We discuss and describe in detail these layers in the following sub-sections.

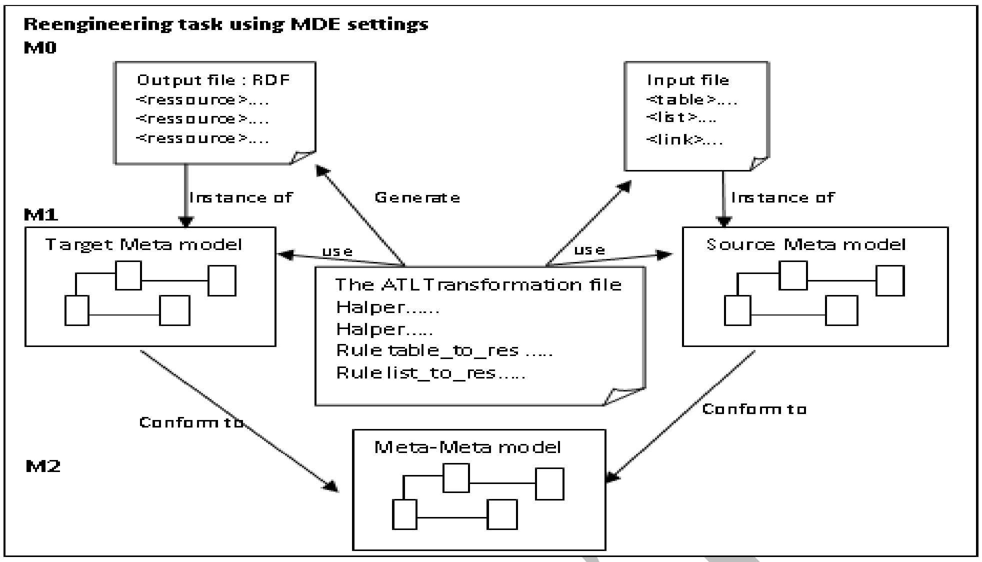


Figure 8. Model-based Reengineering process

4.2.1. Layer M0

In layer M0 corresponding to the real world, there are two files: the input file and the output or the resulting file:

- The first one represents the file obtained from the preprocessing step. It includes data that we wish to transform into semantic data. This file is a set of tags which are defined in the source meta-model, so it constitutes its instance. These tags transport the data that should be transformed into RDF triples.
- The second file is the output one that is obtained by applying the transformation rules on the input file. Both of them contain the same data but in different representations. With RDF as output, data become accessible and understandable, and therefore, machine-exploitable.

4.2.2. Layer M1

It is the meta-model layer. In our system, this layer contains two meta-models: source and target, and the file that contains transformation rules.

The meta-models are written in Ecore language [13], which is a meta-modeling language based on notations quite similar to those of the UML class diagram. In fact, Ecore is an Eclipse implementation of the standard MOF (Meta-Object Facility), since Ecore is defined by itself.

Ecore is the kernel of EMF [24] (Eclipse Modeling Framework) which enables users to build design languages easily.

The transformation rules file is written in ATL language [13]. ATL is a model transformation language based on a concrete textual syntax. ATL provides a mean to match each element from the source meta-model with an element of the target meta-model.

4.2.2.1. Source meta-model (SMM)

Figure 9 shows our source meta-model, through which we defined a set of tags that allows us to write the input file. This meta-model represents the design of HTML elements that contain data. We have conceptualized all the necessary HTML elements, in particular lists, tables, and links, because they often contain more data. As shown in Figure 9, for an HTML table, it can be observed that the element table contains a set of TR elements which contains a set of the TD elements. Note that SMM is written in Ecore language.

Referring to the running example, Figure 10 shows how to write a data list using the tags defined by the source meta-model, precisely, writing the input file in the adaptation step (pre-processing)

4.2.2.2. Target Meta-model (TMM)

We defined in this meta-model only the RDF elements that we need.

As shown in Figure 11, we see that TMM provides a set of RDF elements. Note that this TMM is extensible to cover other RDF elements. Using this meta-model as output allows us to create an RDF file from the input data. Note that TMM is written in Ecore language.

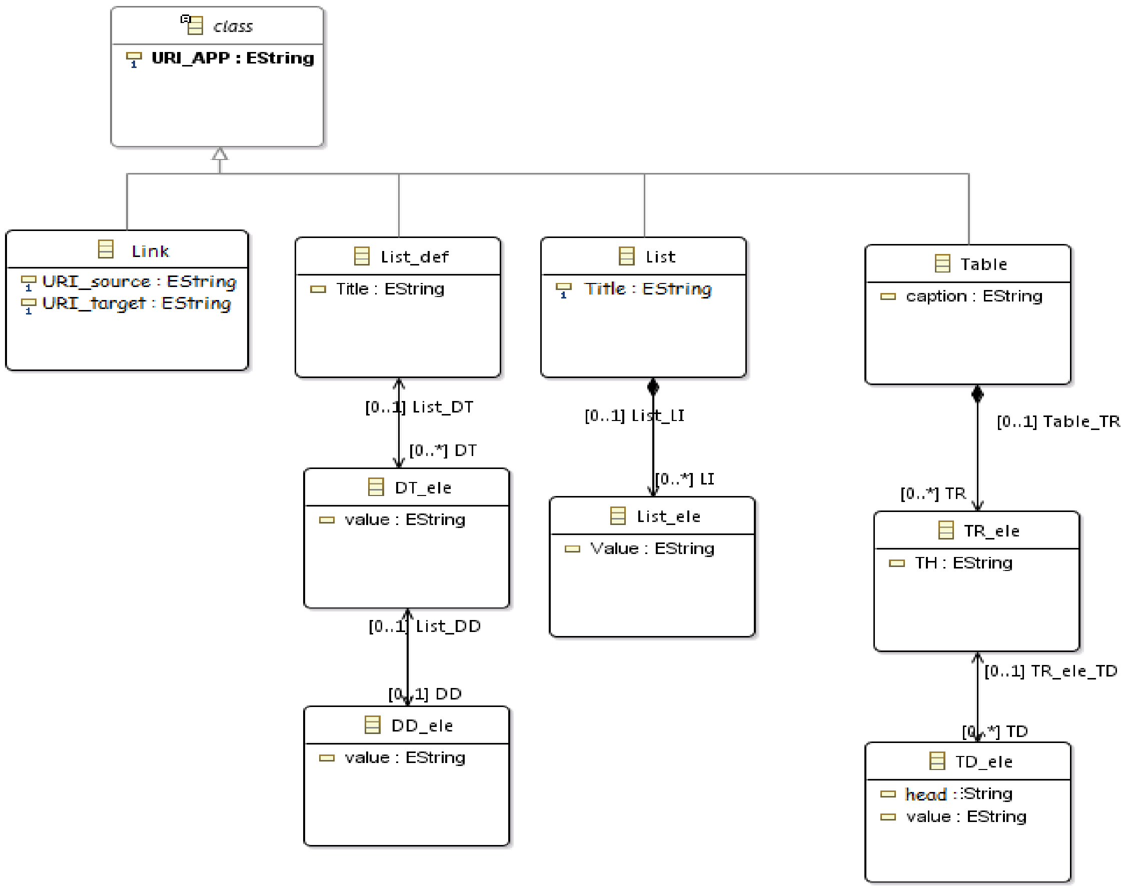
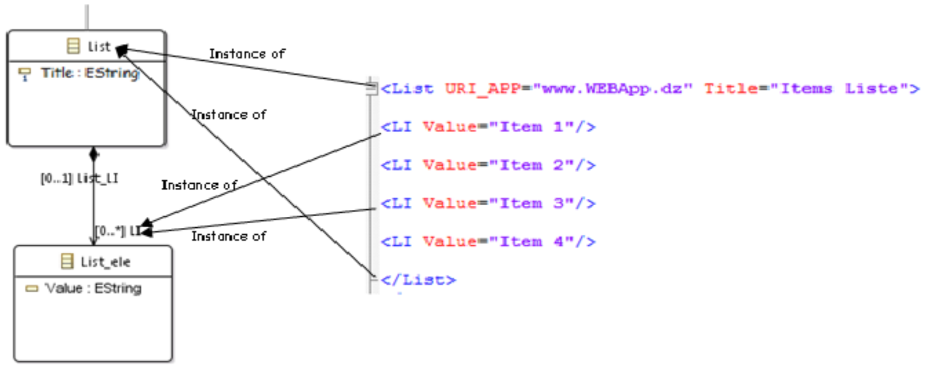


Figure 9. The Source Meta-Model (SMM)



```
<List URI_APP="www.WEBApp.dz" Title="Items Liste">
<LI Value="Item 1"/>
<LI Value="Item 2"/>
<LI Value="Item 3"/>
<LI Value="Item 4"/>
</List>
```

Figure 10. Data list written in SMM tags.

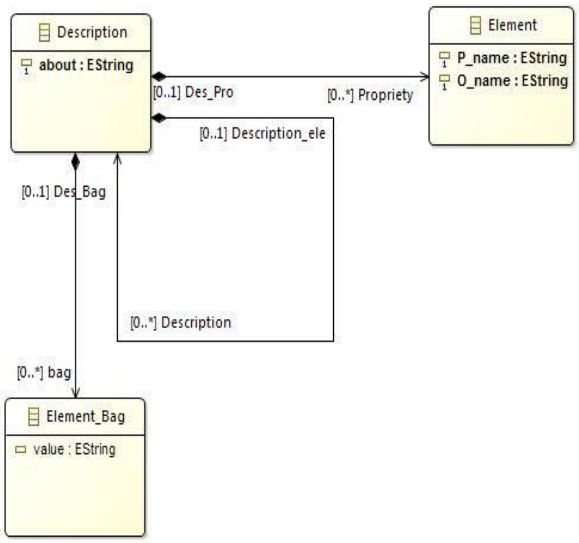


Figure 11. The Target Meta-Model (TMM).

4.2.2.3. Transformation rules

The transformation rules are expressed in ATL language [13]. So, they formed an ATL file. These rules allow matching SMM elements to TMM elements. Since, we would like to transform only HTML elements (tables, links, ordered lists and definition lists) that contain data, we have to define four rules corresponding to the HTML elements.

The transformation process takes as input three files SMM, TMM and the input file resulting from the preprocessing task (and which conforms to SMM), to generate the target RDF file.

Rule 1: Listing 1 is the rule that allows transforming table into an imbricate RDF resource.

The element *Table* becomes a resource, which has the *Table.URI_APP* as a subject. The predicate is the concatenation of the *Table.URI_APP* with the attribute *Table.Caption*. Objects are resources, built as follows:

- Subjects are the concatenation of *Table.URI_APP* with *Table.Caption* and the attribute *TH*.
- Predicates are the concatenation of *Table.URI_APP* with *TD.head*.
- Objects are values of the attribute *TD.Value*.

```
rule Table2Description {
from  s: MMS!Table
to     t: MMC!Description (
                about <- s.URI_APP,
                Propriety<-Set{p},
                Description<-Set{q}),
        p: MMC!Element(
                P_name<-s.predicat_Table,
                O_name<-'nested ressource'),
       q: distinct MMC!Description
                foreach(i in (s.getTH()))(
            about<-s.predicat_Table+'/'+i,
            Propriety<-f),
       f:  distinct MMC!Element
            foreach (n in s.getTDValues()) (
             P_name<-s.getTDtete().toString(),
            O_name<-n.toString())
}
```

Listing 1. The rule of transforming Table into RDF resource (expressed in ATL language).

Rule 2: Listing 2 is the rule that allows transforming ordered list into a bag RDF resource. The element List becomes a resource that has:

- Subject is *List.URI_APP*.
- Predicate is the concatenation of *List.URI_AP* with *List.Title*.
- Object is an RDF bag of the values list obtained from *LI.Value*.

```
rule List2Description {
from   s: MMS!List
to      t: MMC!Description (
                about <- s.URI_APP,
                Propriety<-Set {p},
                bag<-Set{q}),
        p: MMC!Element(
                P_name<-s.predicat_list)  ,
        q: distinct MMC!Element_Bag
                foreach(p in (s.getLiValue()))
                        (value<-p)
}
```

Listing 2. Rule of transforming ordered List into RDF resource (expressed in ATL language).

Rule 3: Listing 3 is the rule that allows transforming a definition List into an imbricate RDF resource.

The element *List_Def* becomes a resource which has:

- Subject is *List_Def.URI_APP*.
- Predicate is the concatenation of *List_Def.URI_APP* with *List_Def.Title*.
- Object is a blank node, having:
  - Predicates are the concatenation of *List_Def.URI_APP*, *List_Def.Title* and *DT.value*.
  - Objects are the values of *DD.Value*.

```
rule ListDef2Description {
from   s: MMS!List_def
to      t: MMC!Description (
                about <- s.URI_APP,
                Propriety<-Set{p},
                Description<-Set{q}),
        p: MMC!Element(
                P_name<-s.predicat_list_def,
                O_name<-'NB'),
        q:  MMC!Description (
              about<-'NB',
                Propriety<-f),
        f:  distinct MMC!Element
                foreach(n in (s.getDTValues()))(
             P_name<-s.predicat_list_def+'/'+n,
             O_name<-s.getDDValue)
}
```

Listing 3. The rule of transforming definition List into RDF resource (expressed in ATL language).

Rule 4: Listing 4 is the rule that allows transforming a Link into RDF resource.

The element *Link* becomes a resource which has:

- Subject is *Link.URI_APP*.
- Predicate is the concatenation of *Link.URI_APP* with *URI_source*.
- Object is the value of the attribute *URI_target*.

```
rule LinK2Description {
from    s: MMS!LinK
to      t: MMC!Description (
             about <- s.URI_APP,
             Propriety<-Set {p}),
p: MMC!Element(
             P_name<-s.predicat_link,
             O_name<-s.URI_target)
}
```

Listing 4. The rule of transforming Link into RDF resource (expressed in ATL language).

Table 1 summarizes the transformation rules presented above. It has four main columns: Rule number, Source elements, Target elements and Matching. The first column indicates the number of the rules. The second and third columns are divided into two sub-columns: Element and Attribute; Element to present the element name of the source or target meta-model, and the column Attribute to list the attributes of the current element. The last column (Matching) shows the correspondence between the elements of the source and target meta-model.

4.2.3 Layer M2

M2 is the layer of the meta-meta-model. In our case, it is the Ecore meta-meta-model language. It is defined in EMF Framework [24]. At this level, it only remains to launch the transformation process to have the output RDF file.

Table 1. Mapping SMM elements to TMM elements.

Rule number	Source Element		Target Element		Matching
	Element	Attribute	Element	Attribute	
1	Table	URI_APP	Description	About	About=URI_APP
		Caption	Propriety	P_name	P_name=URI_APP+caption
				O_name	O_name="nested resources"
	TR	TH	Description	About	About=URI_APP+ caption+ TH
	TD	Head	Propriety	P_name	P_name=URI_APP+ head
		Value		O_name	O_name=value
2	List	URI_APP	Description	About	About=URI_APP
		Title	Propriety	P_name	P_name=URI_APP+Title
	LI	Value		O_name	O_name=null
			Bag	Value	Bag.Value=LI.Value
3	List_def	URI_APP	Description	About	About=URI_APP
		Title	Propriety	P_name	P_name=URI_APP+ Title
				O_name	O_name=”NB”
	DT	Value	Description	About	About=”NB”
			Propriety	P_name	P_name=URI_APP+Title+DT.value
	DD	Value		O_name	0_name=DD.Value
4	Link	URI_APP	Description	About	About=URI_APP
		URI_source	Propriety	P_name	P_name= URI_APP + URI_source
		URI_target		O_name	0_name=URI_target

4.3. Refinement Step

This task (Figure 12) aims to automatically refine the obtained RDF file from the previous step, and it creates an RDF graph corresponding to the refined file. It checks the transformation output file and builds an RDF file in conformity with the RDF grammar defined by W3C consortium.

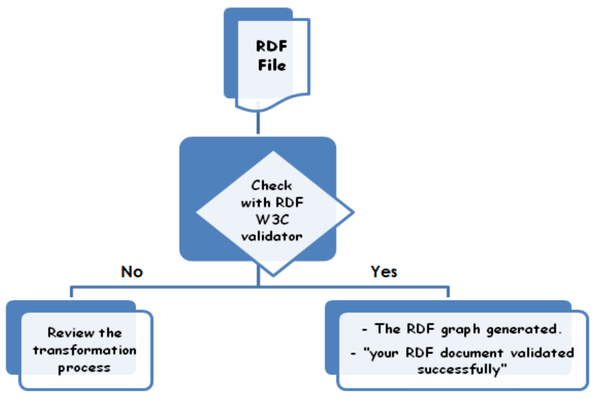

RDF File
Check with RDF W3C validator
No
Yes
Review the transformation process
- The RDF graph generated.
- "your RDF document validated successfully"

Figure12. The Refinement task.

The result of the transformation process is validated by an RDF Validator (we have used the Validator tool provided by the W3C [22]). In case of detecting an error during the validation, the transformation process needs to be reviewed. Otherwise, the W3C Validator provides an RDF graph, and gives to users the following mention «Your RDF document validated successfully. »

Let us take again the running example. The output file obtained at the end of the transformation process is given in Listing 5. This file will be refined in order to construct an RDF file conforms to all W3C rules concerning the RDF language.

```
<?xml version="1.0"  encoding="ISO-8859-1"?>
<xmi:XMI  xmi:version="2.0"
xmlns:xmi="http://www.omg.org/XMI"
xmlns="">
  <Description xmi:version="2.0"
  xmlns:xmi="http://www.omg.org/XMI"
  xmlns=""  about="www.WEBApp.dz">
     <Propriety P_name="www.WEBApp.dz/Items
     Liste"/>
     <bag value="Item  1"/>
     <bag value="Item  2"/>
     <bag value="Item  3"/>
     <bag value="Item  4"/>
  </Description>
</xmi:XMI>
```

Listing 5. The generated RDF file.

Listing 6 displayed the RDF refined file of the running example validated by W3C Validator.

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-
rdf-syntax-ns#"
xmlns:j.0="http://www.WEBApp.dz/Items  ">
<rdf:Description
rdf:about="http://www.WEBApp.dz">
     <j.0:Liste>
     <rdf:Bag>
       <rdf:li>Item  1</rdf:li>
       <rdf:li>Item  2</rdf:li>
       <rdf:li>Item  3</rdf:li>
       <rdf:li>Item  4</rdf:li>
</rdf:Bag>
     </j.0:Liste>
</rdf:Description>
</rdf:RDF>
```

Listing 6.The refined RDF file.

5. Evaluation

5.1. The Implemented Tool

For our experiments, we implemented, with eclipse environment [9], a tool called HTML2RDF. We use Eclipse Modeling Framework [24] (EMF) to describe meta-models, and the transformation language (ATL) to write the transformation rules. The Refinement task is also implemented in Java language.

For evaluating the effectiveness of our tool, we choose some samples of infobox from Wikipedia, and we create for each sample its RDF document that represents infobox data in a collection of triples. Then, we generate an RDF graph, and we validate the obtained result using RDF W3C Validator [22]. We obtain for each sample the same message "Your RDF document validated successfully".

Infoboxes are extracted as HTML tables. We apply a particular preprocessing that deletes undesirable tags like <sup>, <abbr>, <time>,... and we preserve only useful ones like <table>, <td>, <tr> and <a>, tags that contain data. Note that all the <TD> elements that contain multi-values are transformed into a list. The multi-values are separated by comma or <br> tag. The remaining <TD> elements (containing a single value) formed the table.

5.2. Experimentations and Discussions

The chosen samples from Wikipedia are Africa, Algeria, Europe, Tim Berners-Lee, Edgar Frank Codd, MalcomX and Microsoft. Table 2 shows for each sample the evaluation result with the recall, precision, and F-Measure metrics [8].

There are three values *ALL*, *CORRECT* and *EXPERT* used to define the evaluation values. *ALL* corresponds to the number of triples obtained by the RDF W3C Validator. *CORRECT* is the number of the triples that contain information after deleting the triples containing blank node. *EXPERT* is the number of triples that should be returned by our tool (calculated from Wikipedia page).

The expression of recall and precision are given in the following equations (1, 2):

$$precision = \frac{the\ number\ of\ correct\ triples}{the\ number\ of\ all\ obtained\ triples} \qquad (1)$$

$$recall = \frac{the\ number\ of\ correct\ triples}{the\ number\ of\ triples\ that\ should\ be\ returned\ (EXPERT)} \qquad (2)$$

The F-measure combines precision and recall in harmonic metric. The traditional F-measure or balanced F-score is given by:

$$F - measure = 2 * \frac{precision*recall}{precision+recall} \qquad (3)$$

Figure 13 illustrates the three measure recall, precision and F-measure as histograms.

The best value for all measures is 1; it means that the obtained triples by our tool are the same given by the expert. So we accept only the result that has more than 0.8 for F-measure as a good result.

Table 2. Evaluation Results.

RDF triples/sample	Africa	Algeria	Europe	Tim Berners-Lee	E. Codd	Malcom_X	Microsoft
ALL	63	70	70	20	27	44	85
CORRECT	57	48	61	12	17	30	55
EXPERT	56	43	59	14	17	29	49
Recall	1,02	1,12	1,03	0,86	1,00	1,03	1,12
Precision	0,90	0,69	0,87	0,60	0,63	0,68	0,65
F-measure	0,96	0,85	0,95	0,71	0,77	0,82	0,82

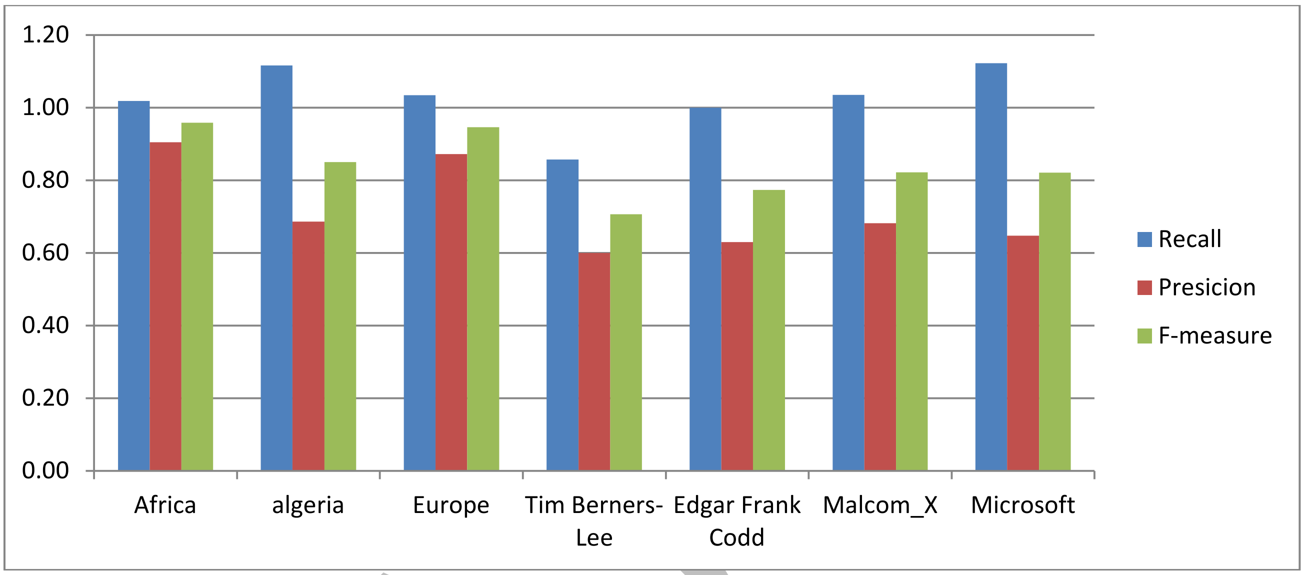


Figure 13. Evaluation results.

We see that we have more than 0.8 for all samples, except the two samples Tim Berners-Lee and Edgar Frank Codd. With an average (for all samples) equal to 0.84, we can say that our system gives encouraging results.

For some samples, values were less than 0.8, such as Tim Berners-Lee and Edgar Frank Codd. These low values are related to the existence of many blank nodes in the result, which are counted as resources, and that's why we have a high number of resources (in ALL value) compared to the number of resources given by the expert. Therefore, we obtained low values of F-measure.

HTML2RDF has two means goals: i) transforming the data stored in HTML pages (coming from database) into RDF format. ii) Transforming the classical Web applications towards semantic ones

6. Related Work

Lehmann et al. present in [15] a DBpedia project that allows mapping the data available on Wikipedia into RDF. They proposed a framework to extract and convert the infoboxes of Wikipedia pages, because they contain the most important information on the top right-hand side of a document in HTML table as pairs (attribute-value). We note that this project is specific to Wikipedia and does not give a generic approach to the other Web applications.

Nagy et al. propose in [19] and [10] a method to generate relational tables from HTML tables. In [19] they used the Header Paths technique, and in [10] they proposed a complementary method to transform relational tables into RDF triples. This approach consists in applying factoring process on HTML tables, therefore the rules used to transform HTML table into RDF triples depend on this process. Note that this approach does not respect the W3C standard rules, presented in [3].

Scharffe et al. present in [23] a project called Datalift. It is a framework for publishing data on the semantic Web in RDF format. These data may be available in different formats, such as CSV, XML and DB. The major problem of this project is their use; to understand how this project works, you must have broad knowledge in the field of semantic Web.

Perez-Castillo et al. present in [21] a re-engineering process, called Preciso, to transform database into Web Services. This process is based on ADM and it consists of four steps: in the first step, the source model is extracted from the database; this source model is in conformity with their SQL-92 meta-model. The second step transforms the source (SQL-92 model) into object model. The third step allows generating WSDL target model which is in conformity with WSDL meta-model [7], and finally generates the basic code of Web service from the object model and WSDL model. This approach is supported by a tool.

Trias et al. present in [25] an ADM-based migration method to automatically transform the traditional Web applications into CMS-based Web applications. This method is based on ADM, and its implementation relies on two artifacts: the first one is a modeling language called ASTM PHP language, which allows defining a model from PHP code. The second artifact is the transformation rules, which generate a target KDM model from ASTM PHP source model. This method is also supported by a tool.

Naima et al. present in [20] a tool called MedPeer, which is "heterogeneous and distributed data" management system that works in peer to peer environment. The main function of this system is to generate OWL ontology from relational database.

Munoz et al. present in [18] an approach to build RDF triples from facts extracted from Wikipedia tables. It relies on a knowledge base of linked data to identify the relationships existing between the entities found in Wikipedia tables.

Arenas et al. present in [3] an approach which allows mapping directly the schema and data of a relational database towards RDF graph, called also a direct graph. The proposed algorithm builds a graph of relative IRIs and uses it with other information, such as primary and foreign keys to achieve the transformation process and generate an RDF graph.

Konstantinou et al. propose in [14] a modular approach to transform metadata, saved in a relational database of the digital library system, into RDF graph. It uses a mapping engine to do this transformation. Note that this approach does not support intelligent queries, and it needs an implementation of the information system.

Mulwad et al. present in [17] and [16] an approach to transform the table contents into RDF. In [17] they combine the resolution and discovery processes to extract entities and relationships. In [16], they propose an extended work of the previous approach to maintain the sense of tables.

7. Conclusion & Perspectives

The Semantic Web is a World Wide Web whose data are accessible to users and machines. Therefore, the programs became able to do well-defined operations on data, and deduce new information from the existing data. The semantic Web offers an RDF language to express data in a manner to make them accessible, and give them a sense. However, in the traditional Web, the data are found either in databases or HTML documents; in both cases they are less exploitable by machines and they haven't sense. To make them in a semantic format we should transform Web applications into semantic ones.

In order to resolve this problem, we have proposed a model-based approach which relies on MDE settings. The proposed approach consists of three steps. The first one is data preparation (preprocessing) which aims to build an input file that conforms to the source meta-model. The second part is the transformation, in which we launch the transformation process to obtain an RDF output file that is in conformity with their target meta-model. The last step is the refinement of the output file in order to get well-formed RDF files.
The obtained RDF files are checked via the W3C Validator. Note that our approach is supported by HTML2RDF tool.

For showing the effectiveness of the proposed approach, we transform some infoboxes of the Wikipedia pages into RDF documents by using the HTML2RDF tool. These RDF documents are checked via the W3C Validator and evaluated with the F-measure metric that gave an average of 0.84, which is an encouraging result.

As a future work, we will carry to process and transform unstructured data, like texts, towards a RDF Linked Data by applying natural language preprocessing techniques.

References

[1] Ardjani F., Bouchiha D., Mimoun M., “Ontology-alignment techniques: Survey and analysis”, *Internation^{2}al Journal of Modern Education and Computer Science (IJMECS)*, vol. 7, no. 11, pp. 67–78, 2015.

[2] Ardjani F., Bouchiha D., Mimoun M., “An approach for discovering and maintaining links in rdf linked data”, *International Journal of Modern Education and Computer Science(IJMECS)*, vol. 9, no. 3,pp. 56–63 ,2017.

[3] Arenas M., Bertails A., Prud’hommeaux E., Sequeda J., “A direct mapping of relational data to rdf , *W3C Recommendation (2012)*, Available at: URL http://www.w3.org/TR/2012/REC-rdb-direct-mapping-20120927/ (accessed: 12 December 2017).

[4] Berners-Lee T., “Linked data - design issues (2006)”, available at: URL http://www.w3.org/DesignIssues/LinkedData.html (accessed: 12 December 2017)

[5] Bezivin J., “Model driven engineering: An emerging technical space”, *Generative and Transformational Techniques in Software Engineering. GTTSE 2005. Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, vol. 4143, pp. 36–64, 2006.

[6] Cafarella M. J., Halevy A. Y., Wang D. Z., Wu E. and Zhang Y., "Web tables: exploring the power of tables on the Web", *proceedings of the VLDB Endowment*, Vol. 1, no. 1, pp. 538–549, 2008.

[7] Christensen E., Curbera F., Meredith G., Weerawarana S.," Web service definition language (wsdl) 1.1", Technical report (W3C), 2001. Available at: URL https://www.w3.org/TR/2001/NOTE-wsdl-20010315 (accessed: 12 December 2017).

[8] David M., Powers W., "Evaluation: From precision, recall and f-factor to roc informedness, markedness and correlation". *Technical Report SIE-07-001 2,* pp. 37–63, School of Informatics and Engineering, Flinders University of South Australia-Adelaide-Australia, 2007.

[9] Eclipse Foundation, "*Eclipse luna release*". Available at: URL https://www.eclipse.org/downloads/packages/release/luna/sr2 (accessed: 12 December 2017).

[10] Embley D. W., Krishnamoorthy M., Nagy G. and Seth S., "Factoring Web tables", *Modern Approaches in Applied Intelligence. IEA/AIE 2011. Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, vol. 6703, pp. 253–263, 2011.

[11] Herman I., "Tutorial in semantic Web (2012)", available at URL: https://www.w3.org/2005/Talks/1214-Trento-IH/#(160) (accessed: 12 December 2017).

[12] Jeffrey T., Pollock M., *Semantic Web For Dummies*, John Wiley and Sons, Inc., Hoboken, New Jersey, 2009.

[13] Jouault F., "Atl: A model transformation too", *Science of Computer Programming,* vol. 72, no.1-2, pp. 31–39, 2008.

[14] Konstantinou N., Spanos D. E., Houssos N., Mitrou N., "Exposing scholarly information as linked open data: Rdfizing dspace contents", *The Electronic Library*, vol. 32, no. 6, pp. 834–851,2014.

[15] Lehmann J., Isele R., Jakob M., Jentzsch A., Kontokostas D., Mendes P. N., Hellmann S., Morsey M., van Kleef P., Auer S. and Bizer C., "Dbpedia a large scale, multilingual knowledge base extracted from Wikipedia", *Semantic Web Journal*, vol. 6, no. 2, pp. 167–195, 2014.

[16] Mulwad V., Finin T., Joshi A.,"Semantic Message Passing for Generating Linked Data from Tables", Proceedings *of the 12th International Semantic Web Conference*, Springer, Berlin, Heidelberg, vol. 8218, pp 363-378, 2013.

[17] Mulwad V., Finin T., Syed Z., Joshi A., "T2LD: Interpreting and Representing Tables as Linked Data", Proceedings of the Poster and Demonstration Session at the 9th International Semantic Web Conf*erence*, CEUR Workshop Proceedings, vol. 658, pp. 25-28, Shanghai, China,2010.

[18] Munoz E., Hogan A., Mileo A., "Using linked data to mine rdf from wikipedia tables", *proceedings of the 7th ACM international conference on Web search and data mining (WSDM 14)*, pp. 533–542, New York, USA, 2014.

[19] Nagy G., Embley D. W., Machado S., Seth S., Jin D. and Krishnamoorthy M., "Data extraction from Web tables: the devil is in the details", *International Conference on Document Analysis and Recognition (ICDAR 2011),* Beijing, China, 2011.

[20] Naima S., Ougouti, Belbachir H., Amghar Y., "A new owl2 based approach for relational database description", *International Journal of Information Technology and Computer Science(IJITCS),*vol. 7, no. 1, pp. 48–53,2015.

[21] Perez-Castillo R., Garcia-Rodriguez de Guzman I., Caballero I., Polo M. and Piattini M., "Preciso: A reverse engineering tool to discover Web services from relational databases", *16th Working Conference on Reverse Engineering (WCRE)*, pp. 309–310, Lille, France,2009.

[22] Prud'hommeaux E., "*Validation service (2006)*". Available at: URL https://www.w3.org/RDF/Validator/ (accessed: 12 December 2017).

[23] Scharffe F., Atemezing G., Troncy R., Gandon F., Villata S., Bucher B., Hamdi F., Bihanic L., Képéklian G., Cotton F., Euzenat J., Fan Z., Vandenbussche P. and Vatant B., "Enabling linked data publication with the datalift platform", *proceedings of AAAI workshop on semantic cities*, Toronto, Canada, 2012.

[24] Steinberg D., Budinsky F., Paternostro M. and Merks E., *Eclipse Modeling Framework: A Developer's Guid*, Addison-Wesley Professional. Part of the Eclipse Series, 2008.

[25] Trias F., Valeria de Castro, Lopez-Sanzand M., Marcos E., "Migrating traditional Web applications to cms-based Web applications", *Electronic Notes in Theoretical Computer Science,* vol 314, pp. 23–44, 2015.

Benamar Bouougada received his engineering degree in computer science and M. Sc. in Computer Science from the University of Mascara, Algeria, in 2007 and 2011 respectively. He joined the Department of Mathematics and Computer Science, University Center of Naama, Algeria in 2012, as Lecturer. His research interests include semantic Web, ontology alignment, and software engineering.

Djelloul Bouchiha received his Engineer degree in computer science from Sidi Bel Abbes University, Algeria, in 2002, and M. Sc. in computer science from Sidi Bel Abbes University, Algeria, in 2005, and Ph. D. in 2011. Between 2005 and 2010, he joined the Department of Computer Science, Saida University, Algeria, as a Lecturer. He became an Associate Professor since September 2013. Currently, he is a Full Professor at the University Center of Naama. His research interests include semantic Web services, Web reverse-engineering, ontology engineering, knowledge management and information systems.

Yassine Ouhammou is an associate professor at National Engineering School for Mechanics and Aerotechnics (ISAE-ENSMA) in France. He received his PhD in 2013. His research focuses on design and analyses of complex systems including real-time systems and database systems by using the model-driven engineering paradigm.

Mimoun Malki graduated with Engineer degree in computer science from National Institute of Computer Science, Algiers, in 1983. He received his M. Sc. and Ph.D. in computer science from the University of Sidi Bel-Abbes, Algeria, in 1992 and 2002, respectively. He was an Associate Professor in the Department of Computer Science at the University of Sidi Bel-Abbes from 2003 to 2010. Currently, he is a Full Professor at École Supérieure en Informatique, Sidi Bel Abbes, Algeria. He has published more than 50 papers in the fields of Web technologies, ontology and reverse engineering. He is the Head of the LabRI-SBA Laboratory. Currently, he serves as an editorial board member for the International Journal of Web Science. His research interests include databases, information systems interoperability, ontology engineering, Web-based information systems, semantic Web services, Web reengineering, enterprise mash up and cloud computing.