IOP Conf. Series: Journal of Physics: Conf. Series **806** (2017) 012009 doi:10.1088/1742-6596/806/1/012009

Generation of Test Questions from RDF Files Using PYTHON and SPARQL

Assel Omarbekova, Altynbek Sharipbay and Alibek Barlybaev

Faculty Of Information Technologies, L.N Gumilyov Eurasian National University, Astana, Kazakhstan

omarbekova@mail.ru
sharalt@mail.ru
frank-ab@mail.ru

Abstract. This article describes the development of the system for the automatic generation of test questions based on the knowledge base. This work has an applicable nature and provides detailed examples of the development of ontology and implementation the SPARQL queries in RDF-documents. Also it describes implementation of the program generating questions in the Python programming language including the necessary libraries while working with RDF-files.

1. Knowledge base development

The aim of the State Program of Development of Education of the Republic of Kazakhstan for 2011-2020 is to increase the competitiveness of education, human capital development by providing quality education for sustainable economic growth. One of the main directions of the program is the electronic learning «e-learning». The implementation of the State Program of Education Development Program of Kazakhstan for 2011-2020 formulated the following task: for each subject studied in primary and specialized school, interactive and intellectual digital educational resources will be developed.

Therefore, nowadays there is a need of the application of artificial intelligence technology in development of educational resources and in the learning process. Currently, the Research Institute "Artificial Intelligence" of L.N Gumilyov Eurasian National University conducts research in this area, developing a knowledge base on the various subject areas. Let's look at the possibility of automatic generation of test questions from the knowledge base.

RDF is developed for computers, so they could understand and read it. It isn't for people and display RDF description on the Internet. RDF is written on XML (Extended Mark-up Language), and it's also a semantic web activity of W3C.

The ontology «ComputerScience.rdf» was developed. RDFS/RDF classes, also the options, like label, domain, range and comment, and attributes, such as about, description, resource datatype, are used in this document [1].

After designing RDF-document «ComputerScience.rdf», the file should be checked for validity using a web service «RDF Validator» on the web-site «www.w3.org». The system displays a list of triples and graph data model. A detail of the generated graph after successful validation of the document «ComputerScience.rdf» is shown in Figure 1. We can see the subjects «http://learningsparql.com/ns/myExample#Database» and «http://learningsparql.com/ns /myExample#Algorithm», eight predicates and eight objects related to their subjects.

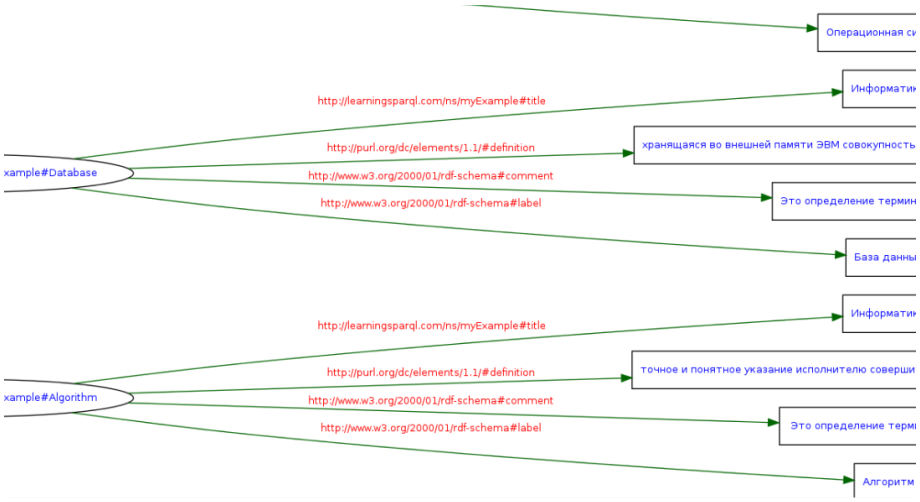

Figure 1. Enlarged part of graph «ComputerScience.rdf»

2. Generation of test questions for the evaluation of students' low-lever skills

The development of test questions for the learning progress evaluation of students requires a serious approach. Tests of high quality are designed carefully so that they are not ambiguous or obscure for the test takers [2]. In the development of test tasks we have investigated various techniques and also deeply studied Bloom's taxonomy. According to the taxonomy, the cognitive skills of pupils are divided into six categories, such as knowledge, comprehension, application, analysis, synthesis and evaluation. The most primitive of the skills are knowledge and comprehension, and the most advanced skills are analysis, synthesis and evaluation. In the development of test tasks, it was decided to generate test questions to test low-level skills in the knowledge and comprehension. Because generating questions for analysis and evaluation a system is needed that is able to make arguments in full. According to Bloom's taxonomy special verbs are defined for the development of test tasks. Some verbs for knowledge and determination, like ‘give the definition’, ‘list’, ‘name’, ‘explain’, etc. With the use of these verbs we can estimate the low-level skills, while verbs, like ‘classify’, ‘locate’ and ‘describe the advantages’ are used to evaluate higher level skills of students.

Program realization starts with installment of IDLE Python 3.5 and «rdflib» library, to develop graphic interface the library «tkinter» is imported into the code. The code of generating test questions of evaluation of low-level skills is shown below.

```
from tkinter import *
import tkinter.messagebox
import rdflib
root = Tk()
def generate():    g = rdflib.Graph()
   g.parse(file_path)
   qres = g.query("""
      PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
      PREFIX ns0: <http://learningsparql.com/ns/addressbook#>
      SELECT ?t ?def
      WHERE    {?a dc:title ?t;       dc:definition ?def. }""")
   global item_arr
   item_arr = []
   for row in qres:       items ="Define the term ' "+str(row[0])+"'"
      item_arr.append(items)
   for item in range(len(item_arr)):
      str_number = str(item+1)
      the_item = str_number+". "+item_arr[item]
```

```
        global label
        label = Label(frame, text = the_item, width="200",height="2", bg="grey", fg="white", font=("Helvetica", 11), justify=LEFT, anchor=W)
        label.pack()
    def openFile():      root.fileName = filedialog.askopenfilename(filetypes=(("RDF files", ".rdf"), ("OWL files", ".owl"), ("All files", "*.*")))
        global file_path
        file_path = root.fileName
        return file_path
    def quit_program():    exit()
    def show_info():      tkinter.messagebox.showinfo('Window title', 'This application generates low order thinking skills questions')
    root.geometry("600x400+300+200")
    root.title("Automated test items generator")
    frame = Frame(root, bg="sky blue", )
    frame.pack()
    menu = Menu(root)
    root.config(menu=menu)
    subMenu = Menu(menu)
    menu.add_cascade(label="File", menu=subMenu)
    subMenu.add_command(label="Open file", command=openFile)
    subMenu.add_separator()
    subMenu.add_command(label="Exit", command=quit_program)
    helpMenu = Menu(menu)
    menu.add_cascade(label="Help", menu=helpMenu)
    helpMenu.add_command(label="Info", command=show_info)
    button1 = Button(frame, text="GENERATE", fg="blue", command=generate)
    button1.pack(side="top", fill='both', expand=True, padx=4, pady=4)
    root.mainloop()
```

The system allows to open any file of rdf and owl type. Choose the file ComputerScience.rdf and click “Generate”. The questions based on a knowledge base will be shown. The form is on Figure 2.

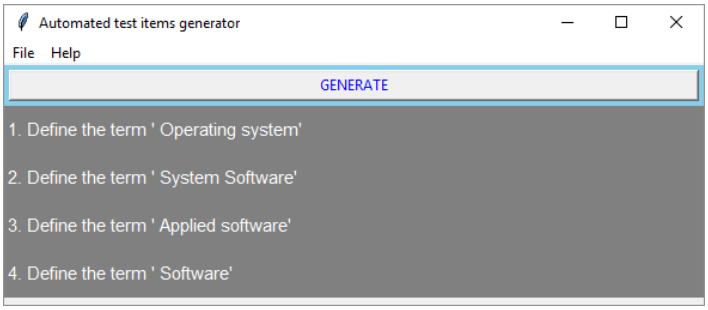

Figure 2. A form with questions testing the knowledge of terms.

3. Generation of test questions to evaluate high-level skills of students

After developing the test questions to evaluate low-level skills, the small owl file under flowers_ontology.owl was developed [3]. A fragment can be seen on Figure 3.

```
<owl:Class rdf:about="http://www.linkeddatatools.com/plants#shrubs">

    <!-- Shrubs is a subclassification of planttype -->

    <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/plants#planttype"/>
    <rdfs:label>Shrubbery</rdfs:label>
    <rdfs:comment>Shrubs, a type of plant which branches from the base.</rdfs:comment>

</owl:Class>

<!-- Individual (Instance) Example RDF Statement -->
<rdf:Description rdf:about="http://www.linkeddatatools.com/plants#magnolia">

    <!-- Magnolia is a type (instance) of the flowers classification -->
    <rdf:type rdf:resource="http://www.linkeddatatools.com/plants#flowers"/>
    <rdfs:label>Magnolia</rdfs:label>
    <rdfs:comment>Magnolia, a type of flower.</rdfs:comment>

</rdf:Description>

<rdf:Description rdf:about="http://www.linkeddatatools.com/plants#rose">

    <!-- Rose is a type (instance) of the flowers classification -->
    <rdf:type rdf:resource="http://www.linkeddatatools.com/plants#flowers"/>
    <rdfs:label>Rose</rdfs:label>
    <rdfs:comment>Rose, a type of flower.</rdfs:comment>

</rdf:Description>

<rdf:Description rdf:about="http://www.linkeddatatools.com/plants#aloe">

```

Figure 3. A file fragment flowers_ontology.owl

Below, on the Figure 4, we can see a generated graph's fragment from the file flowers_ontology.owl

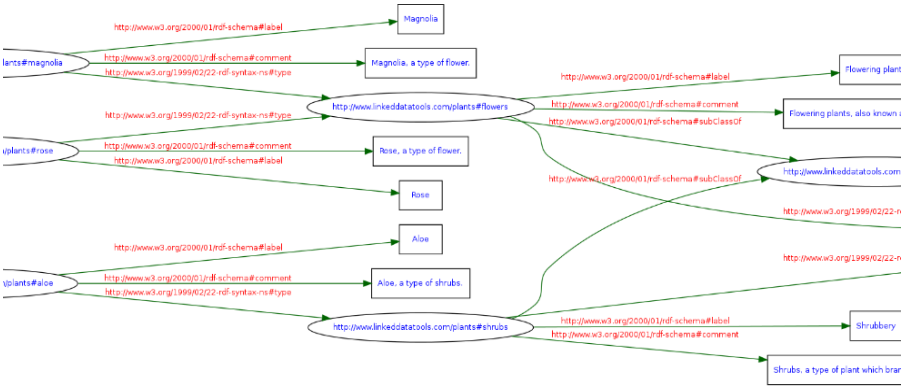

Figure 4. A fragment of generated graph from the file flowers_ontology.owl

The following code allows to generate questions to classify objects.

```
from tkinter import *
import tkinter.messagebox
import rdflib
root = Tk()
def generate():    g = rdflib.Graph()
   g.parse(file_path)# путь к файлу
   qres = g.query("""
   PREFIX   rdfs: <http://www.w3.org/2000/01/rdf-schema#>
   PREFIX    rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
   PREFIX plants: <http://www.linkeddatatools.com/plants#>
   PREFIX     dc: <http://purl.org/dc/elements/1.1/#>
   PREFIX    owl: <http://www.w3.org/2002/07/owl#>
    SELECT ?name
   WHERE
{?entity rdf:type/rdfs:subClassOf* plants:planttype;
        rdfs:label ?name.} LIMIT 5""")
   # cycle to text output  on console
   global item_arr
```

```
        item_arr = []
        for row in qres:        items = str(row[0])
            item_arr.append(items)
        global label
    label = Label(frame, text = " Classify the following terms. To what class they belong?", width="200",height="2", bg="blue", fg="white", font=("Helvetica", 11), justify=LEFT, anchor=W)
        label.pack()
        for item in range(len(item_arr)):        str_number = str(item+1)
            the_item = str_number+". "+item_arr[item]
            label = Label(frame, text = the_item, width="200",height="2", bg="grey", fg="white", font=("Helvetica", 11), justify=LEFT, anchor=W)
            label.pack()
    def openFile():        root.fileName = filedialog.askopenfilename(filetypes=(("RDF files", ".rdf"), ("OWL files", ".owl"), ("All files", "*.*")))
        global file_path
        file_path = root.fileName
        return file_path
    def quit_program():    exit()
    def show_info():       tkinter.messagebox.showinfo('Window title', 'This application generates low order thinking skills questions')
    root.geometry("600x400+300+200")
    root.title("Automated test items generator")
    frame = Frame(root, bg="sky blue", )
    frame.pack()
    menu = Menu(root)
    root.config(menu=menu)
    subMenu = Menu(menu)
    menu.add_cascade(label="File", menu=subMenu)
    subMenu.add_command(label="Open file", command=openFile)
    subMenu.add_separator()
    subMenu.add_command(label="Exit", command=quit_program)
    helpMenu = Menu(menu)
    menu.add_cascade(label="Help", menu=helpMenu)
    helpMenu.add_command(label="Info", command=show_info)
    button1 = Button(frame, text="GENERATE", fg="blue", command=generate)
    button1.pack(side="top", fill='both', expand=True, padx=4, pady=4)
    root.mainloop()
```

Figure 5 illustates a program with graphic interface generating test tasks to classify plants.

After the development of the system, as a result of the testing an error has not been found, and the system regularly generated test items from the knowledge base on the various subject areas. When testing the system several different RDF files were used, in various formats such as, of OWL and TTL. According to test results, rdflib library for the Python programming language, connected to process RDF files, it was not able to work with TTL files. Nevertheless, the system processed RDF and OWL files using the SPARQL query language for RDF documents [4,5].

Also, tests were run and it was attempted to create test questions generated on the basis of the arguments using owlready libraries for python programming language. This library includes Hermit reasoning engine that allows drawing conclusions in respect of certain resources to specific classes. As a result of attempts to generate questions based on logical reasoning the desired results weren’t recieved. Request made to obtain the logical conclusion was made on the owl files, but they did not give the desired results.

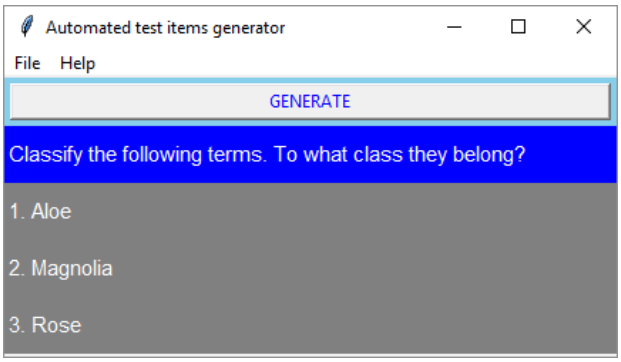

Figure 5. Graphic interface of program with a question

4. Conclusion

Summing up we can draw conclusions regarding the relevance and trends in the development of this direction. Today, all over the world there are a lot of work in the direction of artificial intelligence and intelligent systems based on knowledge [6]. The research shown in this paper provides new ideas and opens up new possibilities of intelligent systems based on knowledge bases for further work and study.

As a result of research the following tasks were solved:

- Development of a knowledge base in RDF format.
- Development of software applications using the IDLE Python to generate test questions.
- Creation of the SPARQL query through IDLE Python in the knowledge base in RDF format.
- Use extracted data from the knowledge base for automatic generation of test questions to test low-level and high-level skills on student knowledge and understanding of the subject area.

The development of systems capable of making inferences from the knowledge base is developing dynamically worldwide. Removing the existing knowledge and generating new knowledge and approving of the existing knowledge base are made by means of specialized engines designed to discuss and draw logical conclusions. The main idea of this work was to extract knowledge from the knowledge base using various latest development tools based on knowledge systems. The system for generating test questions from the knowledge base is one of the steps towards the automation of the evaluation process.

References

[1] DuCharme B 2013 *Learning SPARQL, Querying and Updating with SPARQL* (O`Reilly) pp 3 - 200
[2] Julita B A 2007 *Simplified Guide to Create an Ontology* (ASLab R-2007-004 vol 1)
[3] Negnevitsky M 2002 *Artificial Intelligence – A Guide to Intelligent Systems (*Addison-Wesley Publishing Company. Edinburgh)
[4] William J T 2010 *Artificial Intelligence – Agent Behaviour* p 138
[5] Corby O and Faron-Zucker C 2007 *RDF/SPARQL design pattern for contextual metadata (*Proceedings of the IEEE/WIC/ACM international conference on web intelligence) pp 474-477
[6] Leida M and Chu A 2013 *Distributed SPARQL query answering over RDF data streams (*IEEE international congress on BIG DATA) pp 369-378