Linking E-Learning Ontology Concepts with NLP Algorithms

Dmitry Mouromtsev, Fedor Kozlov, Liubov Kovriguina, Olga Parkhimovich
ITMO University
Saint Petersburg, Russia
d.muromtsev, kozlovfedor, lkovriguina, olya.parkhimovich@gmail.com

***Abstract*—The paper describes applying NLP algorithms to the ontology-based e-learning system. The main goal of the project is to develop a tool creating additional relations between entities based on internal analysis of object property values in the e-learning ontology. Authors present results of automated analysis of links between lecture terms and tests.**

I. Introduction

A. The Ontology-Based E-Learning System

Nowadays reusing educational resources in the Internet becomes one of the most promising approach for e-learning systems development. A good example of using semantics for making education materials reusable and flexible is SlideWiki system [1]. The key feature of an ontology-based e-learning system is the possibility for tutors and students to treat elements of educational content as named objects and named relations between them. These names are understandable both for humans as titles and for the system as types of data. Thus educational materials in the e-learning system thoroughly reflect the structure of education process via the relations between courses, modules, lectures, tests and terms.

In this paper the authors describe an natural language processing module linking task terms with ontology concepts for the ontology-based e-learning system built on top of the Information Workbench platform[1]providing functions to interact with Linked Open Data [2]. Task terms are terms relevant to the course content that occur in tasks to the course. The tasks belong to the test of the course.

B. Description of the ontology of education resources

Information about educational content is stored in the system in RDF(Resource Description Framework) standard. RDF is a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemes differ, and it specifically supports the evolution of schemes over time without requiring all the data consumers to be changed[3].

An original ontology is built on top of top-level ontologies such as AIISO[2], BIBO[3] and MA-ONT[4]. The ontology describes relations between courses, modules, lectures and terms and helps to represent its properties and media content. The most valuable advantage of this ontology is to create direct and indirect interdisciplinary relations between courses [4]. For example physics test "Interference and Coherence" refers also to terms in math ("vector", "vector product"). So if a student cannot pass this test the system advice to repeat not only lecture "Occurrence of Interference" in course "Physics" but also selected lectures on course "Vector algebra". This is example of indirect links between physics and vector algebra via the subject terms "vector" and "vector product".

C. Main problem

A major task in developing and maintaining an educational system is selecting and interlinking relevant materials, e.g. associating the terminology in lectures and tests. When such a link exists between a lecture term and a task term, it is easy to calculate, on the one side, a coverage of tutorials by the checking material (tests) and, on the other side, cases when a term that appear only in some test gets no explanation in tutorials. The main aspect described in the current paper deals with extracting relevant terminology from tests and linking these terms properly with explanatory materials of the system: video lectures, slides, domain terms. Domain terms are instances of the education ontology linked via properties to tasks of the tests/lectures/modules they occur in. To develop an architecture adequate to the set problems, 1) the ontology of tests has been developed, 2) the tests were converted from XML to the RDF standard, 3) the terms were extracted from tasks using NLP and 4) the tasks were linked to the lecture terms via extracted terms.

For designing the described system functionality three course ontology were selected: 1) analytic geometry and linear algebra, 2) graph theory and 3) physics. Each course has modules. Each module has a number of lectures. Material of the lecture is described by a number of terms (annotations objects or a set of keyphrases from user point of view) and media resources. Also each module has tests including from 30 to 100 tasks in a test. Some of lecture's terms must be represented in tasks to be sure a student is able to understand it correctly. An NLP algorithm extracts candidate terms from the text of the task and creates relations between a suitable lecture term and task entities. The relation between these entities is an object property of the ontology class "hasTerm".

II. Ontology of Test

To describe the content of tests a top-level ontology representing test structure has been developed. Developing an ontology based on the structure of tests on physics which are

[1]http://www.fluidops.com/information-workbench/
[2]http://purl.org/vocab/aiiso/schema#.
[3]http://purl.org/ontology/bibo/.
[4]http://www.w3.org/ns/ma-ont#.

used to assess students' knowledge. The test of physics has 20 tasks. Each of them has response options and one right answer. Each version of the test can be made from the database of questions, divided into groups depending on the topic.

The ontology[5] has the following classes: Test, Testing Knowledge Item, Group of Tasks, Task, Answer, Question, Fill-in the Blank, Matching, Multiple Choice, Single Answer, Text Answer, True/False. The classes of the developed ontology are shown at Fig. 1. The main purpose of the developed ontology is to represent structural units of a test and provide automatic task matching by defining semantic relations between tasks and terms[5].

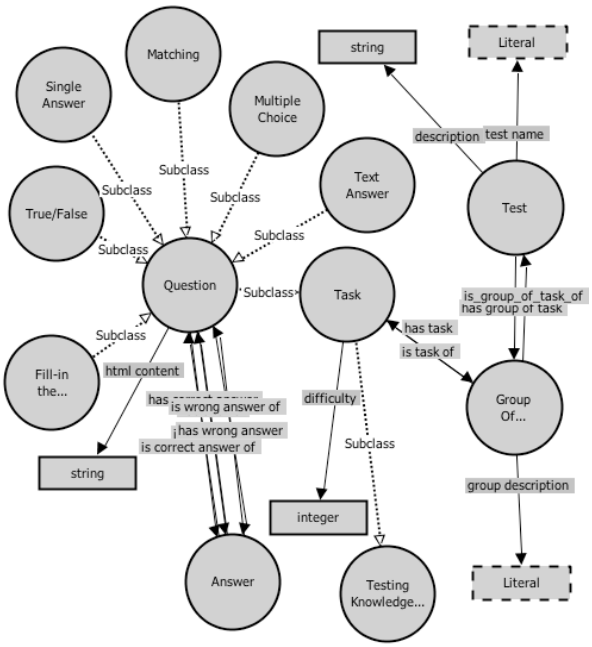

Fig. 1. Main classes of the test ontology

The ontology has class "Test" to store common test characteristics, e.g. its title and description, and class "Testing Knowledge Item" to describe test elements. The class "Testing Knowledge Item" has two subclasses "Group Of Tasks" [6] and "Task". The class "Group Of Tasks" was added to group questions by parameters, e.g. by difficulty. The class "Task" has subclasses "Answer". The class "Question" has subclasses describing question types: "Fill-in the Blank", "Matching", "Multiple Choice", "Single Answer", "Text Answer", and "True/False". The class "Answer" has object properties "is wrong answer of" and "is right answer of". Using this two object properties except one data property "has answer" allow to use one set of answers for many questions.

III. NLP ALGORITHM

Considering the small sample size and pre-set list of lecture terms POS-tag patterns (POS - Part Of Speech) combined with syntax patterns seem to be the most appropriate method to extract terms from the tests [7][8][9]. The same algorithm was used for tests in the Russian language and for the tests translated into English for the demo version. The quality of the algorithm was evaluated on the Russian material only. About ten most typical compound term patterns were used to extract candidate terms (nominal compounds and adjectival compounds). Below are some of them for Russian: "опыт Юнга" <noun in nominative case + anthroponym in genitive case>, "ширина интерференционной полосы" <noun in nominative case + adjective in genitive case + noun in genitive case> and English: "Fresnel biprism", "Poisson light" <anthroponym + noun>, "convexo-plane lens" <adjective + hyphen + adjective + noun>. The patterns were chosen from theoretical works on Russian terminology in different domains.

Due to the rich inflectional system of the Russian language case and number characteristics are specified in POS-tag patterns to extract Russian terms. This helps to avoid extraction of terms belonging to different noun phrases. Some syntactic patterns were also used, because components of a compound term are distant as phrases with coordination ellipsis (a) or belong to different task parts (b):

(a) <adjective + coordinative conjunction (and | or) + adjective + noun>

```
left-handed and right-handed triple
of vectors; coherence length and time;
```

(b) <noun + verb in passive form + adjective>

```
<task>
<question> A matrix with all entries
outside the main diagonal equal
to zero is called
</question>
<answers>
<answer right="no">scalar</answer>
<answer right="yes">triangular</answer>
<answer right="yes">symmetric</answer>
<answer right="no">anti-symmetric</answer>
</answers>
</task>
```

Input plain text was tokenized by spaces and splitted into sentences, then words in it were lemmatized and ascribed morphological information using NooJ dictionaries. NooJ linguistic engine[10] was used to extract terms. NooJ has powerful regular expression corpus search allow to join various POS-patterns in a single grammar that is to be used for querying the text. Dictionaries of lexical entries (for tests and ontology terms) and inflectional grammars were written for the Russian language. To analyze English text for the demo version standard NooJ resources were augmented and reused. NooJ dictionaries allow to combine various linguistic information for the lexical entry, e.g. we tagged anthroponyms (Newton, Fresnel, Poisson, etc.) with a feature "+Anthr" and used it to write a POS-pattern <N+nom+sg><N+gen+sg+Anthr> to extract Russian terms like "бипризма Френеля" ("Fresnel biprism"). Several derivational paradigms for the Russiam morphology were described with NooJ transducers and ascribed to the lexical entries[11]. Assigning derivational paradigms allows to produce a common lemma for the lexical entry and its derivatives, e.g. "coplanar" and "coplanarity" will have common lemma "coplanar". It should be noticed that

[5]http://purl.org/ailab/testontology

NooJ descriptions allow to choose any word of the pair as a derivational basis and e.g. derive "coplanar" from "coplanarity" with a common lemma "coplanarity". NooJ also has a very useful concept of a super-lemma. It allows to link all lexical variants via a canonical form and store them in one equivalence class[12], e.g. in our dictionary a lexical entry "rectangular Cartesian coordinate system" is attached to its acronym "RCCS" (the last is a considered a canonical form) and a query either on acronym or on a compound term matches all the variants. Russian compound candidate terms are transformed to the canonical form (that coincides with a headword in dictionaries) after extraction. E.g. the pattern <adjective + noun> extracts an actual term <feminine adjective in instrumental case + feminine noun in instrumental case>, but lemmatization removes agreement and will produce 2 lemmas: <masculine adjective in nominative case> and <feminine noun in nominative case> whereas the appropriate form of the term is <feminine adjective in nominative case + feminine noun in nominative case>. This doesn't influence the procedure of linking candidate terms to the knowledge base instances, but it is significant for the procedure of validation of missing terms. Transformation to the canonical form is performed by a special algorithm implemented in Python. It uses a special set of canonical templates for Russian, English, etc. languages. The template itself specifies a POS-pattern and a canonical form for this pattern. E.g., if a "N+N+PREP+N" POS-pattern has worked (candidate term = "проекцией<N> вектора<N> на<PREP> ось<N>"), canonical form for the candidate terms is [["nom", "sg"], ["gen", 'sg'], [], []] - "проекция вектора на ось". It means that first noun of the compound terms should be written in nominative case, singular number, second noun should be written in genitive case, singular number and the prepositional phrase should remain as extracted ("[]" means that no grammatical characteristics are specified for it). Below are listed some of templates.

1) "ADJ+ADJ+N": [["", [["nom", "sg"], ["nom", "sg"], ["nom", "sg"]]]],
2) "N+N+ADJ+N": [["", [["nom", "sg"], ["gen"], ["gen"], ["gen"]]]],
3) "N+N+PREP+N": [["", [["nom", "sg"], ["gen", "sg"], [], []]]],
4) "N+N+N+N": [["", [["nom", "sg"], [], [], []]]]

The overall algorithm of term extraction inside the NLP module is the following:

- a plain text is loaded to NooJ that performs its linguistic analysis using provided dictionaries, the output is the plain text with annotations containing morphological and semantic information for every analyzed word (Text Annotation Structure),
- applying queries (that is POS-tag patterns combined with syntactic patterns) stored in a single NooJ grammar file to the Text Annotation Structure, the output is the list of candidate terms,
- candidate terms with annotations are exported to a text file,
- candidate terms are converted to the canonical form using the list of word forms with morphological annotations generated in NooJ.

Table I. EXAMPLE OF TEST ENTITY CONVERSION

The input XML code

```
<test
    module="m_InterferenceAndCoherence"
    module_ns="Phisics"
    uri="TestOfInterferenceAndDiffractionFrenel"
    name="Test Of Interference
        And Diffraction Frenel">
</test>
```

The mapping code

```
<rule
    id="test" nodeBase="//test"
    owlType="learningRu:Test"
    instanceNamespace="openeduTests"
    objectId="{./@uri}"
    objectLabel="{./@name}">
    <objectPropertyMapping
        nodeBase="."
        instanceNamespace="openeduTests"
        value="{./@name}"
        owlProperty="ifmotest:hasGroupOfTasks"
        referredRule="task_group" />
</rule>
```

The output RDF/XML code

```
<rdf:Description
    rdf:about="http://openedu.ifmo.ru/tests/
    TestOfInterferenceAndDiffractionFrenel">
  <rdf:type
    rdf:resource="http://www.semanticweb.org/
    k0shk/ontologies/2013/5/learning#Test"/>
  <label
    xmlns="http://www.w3.org/2000/01/rdf-schema#">
    Test Of Interference And Diffraction Frenel
  </label>
  <hasGroupOfTasks
    xmlns="http://www.semanticweb.org/fedulity
    /ontologies/2014/4/untitled-ontology-13#"
    rdf:resource="http://openedu.ifmo.ru/tests/
    Test_Of_Interference_And_Diffraction_Frenel"/>
</rdf:Description>
```

IV. IMPLEMENTATION

A. Test parsers

To convert test data from XML format to RDF standard the mapping was described. To provide conversion in the system the XMLProvider instance was created. The mapping for the test data conversion was described in the XML format. The mapping allows to automatically convert XML files of the tests to the semantic data in accordance of the test ontology. The XMLProvider uses XPath functions to extract data about objects and properties from the input XML file. The extracted data is converted into the RDF/XML format based on the mapping description The example of the input XML code, the mapping and the output result for the test entity conversion is in table I.

The provider supports periodic updating of data obtained from remote XML sources. In accordance with the designed test ontology the wiki template pages for the test data were created. The wiki template pages are based on the Semantic MediaWiki syntax[13] and stored inside the Information Workbench system. The wiki template pages are automatically

instantiated for resources of some fixed type. The wiki template pages allow to represent semantic data to the user in a convenient form.

User interface of the task with linked terms is shown at Fig. 2.

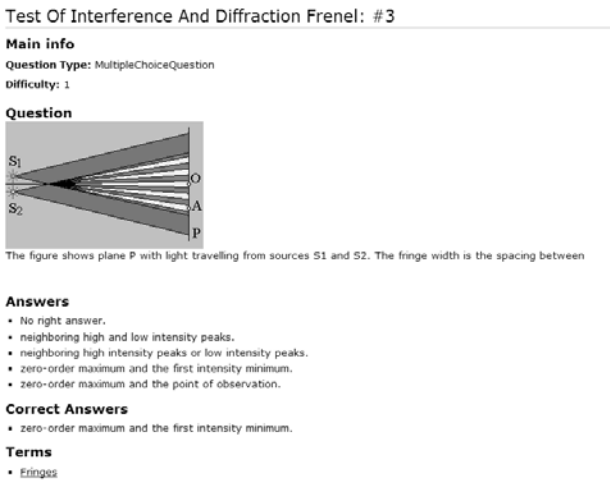

Fig. 2. User interface of the task with linked terms

B. The NLP module

To handle links between system terms and test tasks a new data provider was implemented. The provider supports periodic updating of links. The input of the provider is the URI of the course entity. The provider handles all links between subject terms and test tasks of the input course. The provider is implemented in Java and uses the standard libraries and the Provider SDK of the Information Workbench platform.

The provider is based on the following algorithm:

- the provider collects tasks of the course by using SPARQL queries;
- the provider forms the plain text content for each task by using the information about questions and answers of the task;
- the provider launches NLP procedures in NooJ for the plain text content of the task;
- the provider extracts lemmas of candidate terms from the NooJ output file,
- the provider searches terms in the system to be linked with candidate terms by using SPARQL queries, system terms and candidate terms are linked if they have the same lemma/lemmas;
- the provider creates a link between selected system terms and the task by using the "hasTerm" property.

Example of candidate term extraction and linking from the single task is in table II.

Table III shows the results of the algorithm work that are somehow ambivalent: on one side, 95% of tasks are linked with at least one system term, on the other side, only 50 % of

Table II. EXAMPLE OF THE TERMS EXTRACTION FROM THE SINGLE TASK

Plain text

```
<img src="http://open.ifmo.ru/images/3/31/
     2268451_int019.gif"/>

In Young experiment a thin glass plane is set at the
path of the ray d2, that results in biasing of the
central band to the position initially taken
by the 5th light interference band. Radiating
wavelength is 600 nm, plane refractive index n=1,5.
What is the thickness of the plane?
No right answer. 8,4 7,2 6,0 4,8 3,6
```

The output of the NooJ module

```
1. <LU LEMMA="image" CAT="N" FLX="TABLE" Nb="p">
   images</LU>
2. <LU LEMMA="Young" CAT="N" FLX="TABLE" TYPE="Prop"
   Nb="s">Young</LU>
3. <LU LEMMA="Young" CAT="N" FLX="TABLE" TYPE="Prop"
   Nb="s">Young</LU>
   <LU LEMMA="experiment" CAT="N" FLX="TABLE" Nb="s">
   experiment</LU>
4. <LU LEMMA="experiment" CAT="N" FLX="TABLE" Nb="s">
   experiment</LU>
5. <LU LEMMA="glass" CAT="N" FLX="TABLE" Nb="s">
   glass</LU>
6. <LU LEMMA="glass" CAT="N" FLX="TABLE" Nb="s">
   glass</LU>
   <LU LEMMA="plane" CAT="N" FLX="TABLE" Nb="s">
   plane</LU>
7. <LU LEMMA="plane" CAT="N" FLX="TABLE" Nb="s">
   plane</LU>
8. <LU LEMMA="path" CAT="N" FLX="TABLE" Nb="s">path</LU>
9. <LU LEMMA="ray" CAT="N" FLX="TABLE" Nb="s">ray</LU>
10. <LU LEMMA="central" CAT="A">central</LU>
    <LU LEMMA="band" CAT="N" FLX="TABLE" Nb="s">band</LU>
11. <LU LEMMA="band" CAT="N" FLX="TABLE" Nb="s">band</LU>
12. <LU LEMMA="position" CAT="N" FLX="TABLE" Nb="s">
    position</LU>
13. <LU LEMMA="light" CAT="N" FLX="TABLE" Nb="s">
    light</LU>
14. <LU LEMMA="light" CAT="N" FLX="TABLE" Nb="s">
    light</LU>
    <LU LEMMA="interference" CAT="N" FLX="TABLE" Nb="s">
    interference</LU>
15. <LU LEMMA="interference" CAT="N" FLX="TABLE" Nb="s">
    interference</LU>
16. <LU LEMMA="interference" CAT="N" FLX="TABLE" Nb="s">
    interference</LU>
    <LU LEMMA="fringe" CAT="N" FLX="TABLE" Nb="s">
    fringe</LU>
17. <LU LEMMA="fringe" CAT="N" FLX="TABLE" Nb="s">
    fringe</LU>
18. <LU LEMMA="wavelength" CAT="N" FLX="TABLE" Nb="s">
    wavelength</LU>
19. <LU LEMMA="thickness" CAT="N" FLX="TABLE" Nb="s">
    thickness</LU>
```

URIs of the terms in the system

```
GraphTheory:t_Path
Physics:t_Interference
Physics:t_Light
Physics:t_Wavelength
Physics:t_Fringes
Physics:t_Young's Experiment
```

system terms coincide to the terms of the tasks. The fact that 10% of candidate terms (like "Fresnel diffraction", "Fresnel zones", "Michelson interferometer", "refractive index") should be added to the pool of system terms can be considered as a positive result. To filter out false candidates term validation should be done using outer data sources like DBpedia, etc.

C. Lecture Coverage Analysis

The analysis of lecture coverage by tests is performed inside the module. Both test and lecture entities are associated

Table III. EVALUATION OF THE NLP-MODULE (FOR THE ENGLISH LANGUAGE)

Percent of linked tasks, %	95
Total number of different candidate terms	155
Total number of manually extracted terms	30
Percent of system terms, matched by candidate terms, %	50
Percent of candidate terms, matched by system terms, %	8
Percent of interdisciplinary candidate terms, %	3
Percent of general scientific terms, %	20
Percent of embedded candidate terms, one of which matches a system term, %	29

with the module entity, so the results can be obtained by analyzing terms of the test as well as terms of the lecture. Each module has a separate analytics page that contains widgets, tables and carts. The data of all UI elements on the page are obtained by using SPARQL queries. The system analytics page of the module includes basic statistics and lecture coverage statistics. The basic statistics comprises:

- information about the total number of covered and uncovered terms of the module,
- cover ratio of the module based on the ratio of the number of covered terms among total number of module terms,
- a tag cloud of the most covered terms of the module,
- a table of the test terms not covered by lectures of the module.

User interface of the basic statistics is shown at Fig. 3.

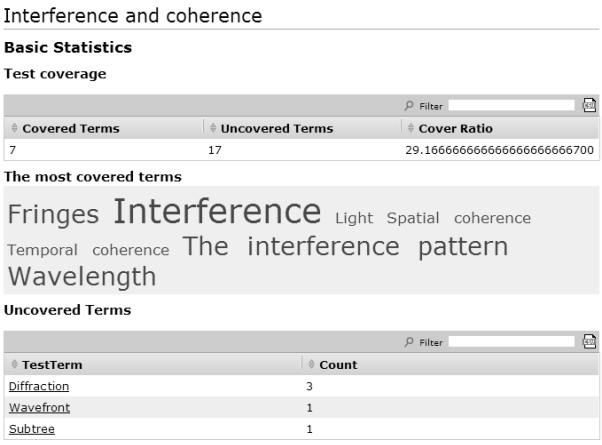

Fig. 3. User interface of the basic statistics

The lecture coverage statistics shows the ratio of covered terms among total number of lecture terms for each lecture of the module. The lecture coverage statistics is represented in a bar chart.

User interface of the lecture coverage statistics is shown at Fig. 4.

The obtained set of analytical data helps to evaluate adequacy of educational content and get a notion what content is ample or needs to be changed or added.

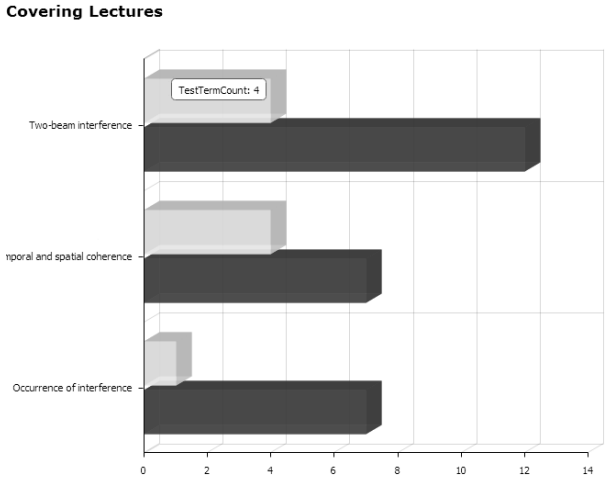

Fig. 4. User interface of the lecture coverage statistics

V. CONCLUSION

The developed modules for the e-learning system provide tutors with a tool to maintain high quality of existing knowledge assessment modules. With these modules tutors could fluently change education resources, content and tests keeping them up-to-date.

Example of tests covering analytics for module "Interference and Coherence" can be found at

```
http://openedu.ifmo.ru:8888/resource/
    Phisics:m_InterferenceAndCoherence?
    analytic=1
```

The source code can be found at

```
https://github.com/ailabitmo/
    linked-learning-solution.
```

Future work for the NLP-module implies describing a set of term periphrases and developing a procedure to choose the best candidate from embedded candidate terms like in: "glass refractive index", "refractive index"; "light source", "monochromatic light source", "point monochromatic light source". The algorithm should also filter out candidate terms that are non-thematic to the course, e.g. if a term "vector" occurs in a task on physics, it should not be marked as a term highly relevant to the course on interference because it is introduced in another course. The idea is that a link is created between a system term and any term that occurred in the task, but terms that do not belong to the topic of the course should not be marked as terms missing in the course. Besides, there is a far more challenging problem: a task may not contain explicitly the term it is intended to checks. Consider the following example:

```
A ladder is 5m long. How far from the
base of a wall should it be placed if
it is to reach 4m up the wall?
Give your answer in metres correct
to 1 decimal place.
```

This task checks understanding of the Pythagorean Theorem, but it contains no explicit information allowing to assign proper keywords to the task. Meanwhile, such tasks are quite numerous. Right now the algorithm fails to process such tasks remaining them unlinked. Elaborating the algorithm to handle cases like this is the work to be done. Term extraction procedure can be also improved for adding parallel texts of tasks. The provider needs to be refined to create test entities in several languages. Linking terms and tasks is valuable to control students knowledge, since it provides the statistics of true and false answers to the question associated with particular terms, that allows to recommend students to repeat a lecture, module or course.

This work was partially financially supported by the Government of Russian Federation, Grant 074-U01.

REFERENCES

[1] Khalili, A., Auer, S., Tarasowa, D., Ermilov, I. "SlideWiki: elicitation and sharing of corporate knowledge using presentations", *In Knowledge Engineering and Knowledge Management*. Springer Berlin Heidelberg. 302-316 (2012)

[2] Haase, P., Schmidt, M., Schwarte, A. "The Information Workbench as a Self-Service Platform for Linked Data Applications", *COLD*, (2011)

[3] Klyne, Graham, and Jeremy J. Carroll. "Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation, 2004.", *World Wide Web Consortium*, URL: http://w3c. org/TR/rdf-concepts, (2004)

[4] Mouromtsev, D., Kozlov, F., Parkhimovich, O., Zelenina, M. "Development of an Ontology-Based E-Learning System", *Knowledge Engineering and the Semantic Web*. Springer Berlin Heidelberg, 273-280 (2013)

[5] Soldatova, L., Mizoguchi, R. "Ontology of test", *Proc. Computers and Advanced Technology in Education*. 173-180 (2003)

[6] Vas, R. "Educational ontology and knowledge testing", *The Electronic Journal of Knowledge Management of 5.1*, 123-130 (2007)

[7] Hulth A. "Improved Automatic Keyword Extraction Given More Linguistic Knowledge", *Proceedings of the 2003 conference on Empirical methods in natural language processing (EMNLP'03)*. P.216-223 (2003)

[8] Khokhlova M.V. "Lexico-syntactic patterns as a tool for extracting lexis of a specialized knowledge domain", (in Russian). *Proceedings of the Annual International Conference "Dialogue"* (2012).

[9] Bolshakova E., Vasilieva N. "Formalizacija leksiko-sintaksicheskoj informacii dlja raspoznavanija reguljarnyh konstrukcij estestvennogo jazyka [Formalizing lexico-syntactic information to extract natural language patterns]", *Programmnye produkty i sistemy [Software and Systems]*. No.4. P.103-106. (2008)

[10] Silberztein M. NooJ for NLP: a linguistic development environment. URL: http://www.NooJ4nlp.net/pages/NooJ.html (2002 - :)

[11] Silberztein M. NooJ Manual [Electronic resource]. P.99. URL: http://www.NooJ4nlp.net/NooJManual.pdf, (2003)

[12] Ibid. P.82.

[13] Krötzsch, M., Vrandečić, D., Völkel, M. "Semantic mediawiki", *The Semantic Web-ISWC 2006*. Springer Berlin Heidelberg, 935-942 (2006)