A Combined Method for E-Learning Ontology Population based on NLP and User Activity Analysis

Dmitry Mouromtsev, Fedor Kozlov, Liubov Kovriguina and Olga Parkhimovich

ITMO University, St. Petersburg, Russia
d.muromtsev@gmail.com, kozlovfedor@gmail.com, lkovriguina@gmail.com,
olya.parkhimovich@gmail.com

Abstract. The paper describes a combined approach to maintaining an E-Learning ontology in dynamic and changing educational environment. The developed NLP algorithm based on morpho-syntactic patterns is applied for terminology extraction from course tasks that allows to interlink extracted terms with the instances of the system's ontology whenever some educational materials are changed. These links are used to gather statistics, evaluate quality of lectures' and tasks' materials, analyse students' answers to the tasks and detect difficult terminology of the course in general (for the teachers) and its understandability in particular (for every student).

Keywords: Semantic Web, Linked Learning, terminology extraction, education, educational ontology population

1 Introduction

Nowadays reusing online educational resources becomes one of the most promising approaches for e-learning systems development. A good example of using semantics to make education materials reusable and flexible is the 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 relations between courses, modules, lectures, tests and terms.

In this paper the authors describe development of two modules of the ontology-based e-learning system built on top of the Information Workbench platform[1] (the latter provides functions to interact with Linked Open Data[2]): a system user module aggregating information about students' activity and an NLP module responsible for terminology extraction and terms' linking to the instances of the system knowledge base.

[1] http://www.fluidops.com/information-workbench/

2 Related Work

Some e-learning and educational systems use ontologies and semantic web. For example, educational ontology which supports the creation of transparent curriculum content and adaptive knowledge testing system were developed by Réka Vas [6]. Another example is the ontology-driven e-learning system for Thai learning environment (O-DEST) [9]. O-DEST contains an ontology for e-learning process which describes teaching methods, learning styles and activities and it is used by teachers and administrative personnel to configure and maintain course materials. Metacademy[2] system is an example of the e-learning system using terminology to link learning resources. Metacademy is a community-driven, open-source platform for experts to construct collaboratively a web of knowledge. Metacademy is the e-learning system, where the content of education (lectures, videos, books) linked through the subject terms.

3 Motivation

A major task in developing and maintaining an educational system is choosing and interlinking relevant materials, e.g. associating terms in lectures and tests. 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. The latter are instances of the education ontology linked via properties to tasks of the tests/lectures/-modules they occur in. When the links are created, any sophisticated statistics can be gathered, e.g. statistics about students' correct/incorrect answers allowing to filter out troublesome terms and topics. The last provides teachers with a mean to improve their lectures. Modules gathering statistics about students' activity in the system are implemented on the front-end applications. Statistics is stored in the knowledge base of students' activity. 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 Semantic Web format, 3) the terms were extracted from tasks using morpho-syntactic patterns, 4) the tasks were linked to the lecture terms via extracted candidate terms, 5) candidate terms that do not match any instance in the system knowledge base were validated via DBpedia[4], 6) the ontology of the e-learning system user has been developed, 7) statistics gathering module has been developed, 8) pages to show statistics of students' answers to the tests have been created.

The described system functionality was designed on the material of three courses: 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 the user's point of view) and media resources. Also, each module has tests including from 30 to 100 tasks in a test. Most part of lecture's terms should be represented in tasks to ensure that a student understood the

[2] http://www.metacademy.org/

lecture 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".

4 Ontology development

4.1 Description of the ontology of education resources

An original ontology is built on top of top-level ontologies such as AIISO[3], BIBO[4] and MA-ONT[5]. The ontology describes relations between courses, modules, lectures and terms and helps to represent its properties and media content. The most outstanding feature of this ontology is its ability to create direct and indirect interdisciplinary relations between courses[3]. E.g., physics test "Interference and Coherence" includes math terms as well ("vector", "vector product"). Thus, if a student can't pass this test, the system advices to repeat not only the lecture "Occurrence of Interference" in the "Physics" course, but also corresponding lectures from the "Vector algebra" course. This is an example of indirect links between physics and vector algebra via the subject terms "vector" and "vector product".

4.2 Ontology of Test

To describe the content of tests a top-level ontology representing test structure has been developed. Top-down approach was used to develop ontologies for the educational system. It was used because developed ontology extended top-level ontology. The ontology[6] 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 in the figure 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]. 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 subclass "Task". The class "Group Of Tasks" [6] 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.

[3] http://purl.org/vocab/aiiso/schema#.
[4] http://purl.org/ontology/bibo/.
[5] http://www.w3.org/ns/ma-ont#.
[6] http://purl.org/ailab/testontology

4.3 Ontology of student activity in the e-learning system

The ontology of student activity[7] is designed to store information about the student's learning process and results. Two top-level ontologies have been used for its development: ontology of test, as described above, and FOAF ontology[8] that describes people and relationships between them.

The classes of the developed ontology are shown in the figure 2. The class "Learning process" was added to store information about actions performed by a student in the system. Students can watch video (subclass "Video"), try to pass the test (subclass "AttemptToPassTest"), learn terms (subclass "Term") and pass a course (subclass "Course"). The ontology also has class "Student" to store information about users and their activity in system. This class is a subclass of class "Person" determined in FOAF ontology. The object properties "enrolled course", "finished course", and "subscribed course" describe relationships between the class "Student" and the class "Course". The class "Learning results" was added to store information about students educational activities and answers. Class "TestElement" contains information about "Task" (class of test ontology) and about student's "Answer" (subclass of class "LearningResults"), which can be correct or incorrect. Set of test elements constitutes attempt to pass test. The properties "timestamp of attempt" and "percent complete of test" allow e-learning system to store information about the time in which an attempt was made and to determine the result of the test. The e-learning system uses the ontology of tests and answers given by the user to build a list of terms that the user knows.

5 NLP algorithm

Considering the small sample size and pre-set list of lecture terms POS-tag patterns combined with syntax patterns seem to be the most appropriate method to extract terms from the tests [7][8][10]. The same algorithm was used for tests in the Russian language and for the tests translated into English for the demo version. 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>.

Due to the rich inflectional system of the Russian language case and number characteristics are specified in POS-tag patterns to extract Russian terms. 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):

[7] http://purl.org/ailab/learningresults
[8] http://www.foaf-project.org

(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>
```

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.

NooJ linguistic engine[11] was used to extract terms. NooJ has powerful regular expression corpus search allowing to join various POS-patterns in a single grammar to query the text. Dictionaries of lexical entries (for tests and ontology terms) and inflectional grammars were written for the Russian language by the authors of the paper. Lexical resources developed for the Russian language cover tasks' vocabulary totally. 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[12]. 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 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[13], 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. 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.

To apply the NLP-algorithm to other domains and languages one needs to compile NooJ lexical resources (dictionaries), write grammars and work out the templates to extract terms.

6 Methods

6.1 Test parsers

To convert test data from XML format to semantic format a mapping was described. To provide conversion in the system an XMLProvider instance was created. The mapping for the test data conversion was described in the XML format. The mapping allows to convert XML files of the tests to the semantic data in accordance of the test ontology automatically. 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 1.

6.2 The NLP module

To map a candidate term to the system term via lemma, system terms were also lemmatized. Each system term has been assigned a text property "lemma" with a label containing the lemma of a term.

To handle links between system terms and test tasks the 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 using SPARQL queries;

Table 1. 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>
```

- the provider forms the plain text content for each task 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 candidate terms from the NooJ output file, the data contain a canonical form and lemma(s) for the candidate term;
- the provider searches terms in the system to linked them with candidate terms by using SPARQL queries, system terms and candidate terms are linked if they have the same lemma(s);

– the provider creates a link between selected system terms and the task by using the "hasTerm" property.

If a word sequence extracted with a morpho-syntactic pattern doesn't match any of the system terms, it becomes a candidate instance to be included to the system as a new system term. Apparently, it is necessary to validate it, e.g., via external sources. We chose DBpedia to validate candidate instances. A candidate instance is considered a new system term if its canonical form (a headword) completely matches to DBpedia instance's property rdfs:label or dbpedia-owl:wikiPageRedirects, otherwise it is considered a false candidate. To avoid false matches results are filtered by the property "dcterms:subject". Validation is considered successful in case one or more DBpedia instances were matched. The validated candidate instance is added to the system as a new candidate term and is linked to the task. It becomes authentic system term after teachers' approval.

An example of querying DBpedia via SPARQL-endpoint is below:

```
SELECT DISTINCT ?term {
    ?term dcterms:subject ?subject .
    VALUES ?subject {
        category:Concepts_in_physics
        category:Physical_optics
        category:Optics}
    {?name_uri dbpedia-owl:wikiPageRedirects ?term ;
        rdfs:label ?label .
    }
    UNION
    { ?term rdfs:label ?label }
    FILTER( STR(?label) = "Diffraction")
}
```

7 Evaluation and Results

Procedures of candidate terms extraction and validation that were described in 5 and 6.2 above have produced the following results in the table 2. Term validation via DBpedia as it is proposed in the paper is merely an idea rather than a technique. Using it in the described straightforward manner, we pursued the aim to remove the bulk of false candidates, not to validate the largest possible number of the candidate terms. Overall, 30 different terms were extracted manually from 20 tasks and 5 times more candidate terms were extracted using POS-patterns. The system contains 24 terms on interference and diffraction in the physics course at the moment.

The obtained results seem rather ambivalent: on the one side, 95% of tasks were linked to at least one term. The leading system term is "Light", that has been linked to 12 tasks. On the other side, 50% of system terms remained unlinked to tasks and about 60% of them demand addition of proper tasks. The

Table 2. Evaluation of the NLP-module (for the English language)

Percent of linked tasks, %	95
Percent of non-linked tasks, %	5
Number of different candidate terms	155
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 candidate terms to be included to the system terms after the validation procedure, %	6
Percent of false candidates, %	86

validation procedure using DBpedia as an external source provided 9 terms to be added as candidate system instances ("wavelength", "coherence", "coherent light", "diffraction", "amplitude", "aperture", "diffraction pattern", "optical path", "optical path length"). We treated all the rest terms (that do not match any system term and failed DBpedia validation procedure) as false candidates. However, actually a few of false candidates are true terms that do not present in chosen categories of DBpedia (Concepts_in_Physics, Optics and Physical_optics), but present in other DBpedia categories (e.g. "Fresnel diffraction", "Fresnel zone" and "Fresnel number" are in the category: "Diffraction", "Michelson interferometer" is in the category "Interferometers"). Some terms have different canonical form in Russian and English, e.g. "Young's interference experiment" (is in DBpedia) reduces to "Young's experiment" in Russian (no term in DBpedia). Thus, developers depend completely upon the data segmentation of the external source. 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.

8 The module of the user statistics collection

The front-end of the ontology-base e-learning system is a lightweight Learning Management System. The front-end is designed to represent educational content conveniently. It also manages user data, user settings and the results of the user's educational process. The front-end handles content administration, restricts access to educational content and supports widgets for video lectures, tests and practices.

The user interface of the front-end test page is shown in fig. 3.

The front-end is implemented in Python[9] and uses the Django Web Framework[10]. Data from the educational content are extracted with SPARQL queries to the Information Workbench SPARQL-endpoint[14]. The SPARQLWrapper Library[11] is used to compile SPARQL queries. When the system user has finished the test, the module gathering user's statistics sends the SPARQL Update Query[15] with user answers to the Information Workbench SPARQL-endpoint. When user statistics is gathered, objects having type "AttemptToPassTest" and user's answers to the test's tasks are created in the system. The object with type "AttemptToPassTest" is bound to hash data of user's e-mail.

8.1 Analysis of user responses to the tests

The data about number of correct/incorrect user's answers allow to compute the knowledge rating for any of the system terms. Using this rating, teachers can determine which terms of the course students know worst of all. The knowledge rating is count by subtracting the number of incorrect answers from the number of correct answers for all tasks which contains this term. This rating is quite simple and is to be replaced by a ranking formula after elaborating a set of features.

Data about user's answers are collected with the following SPARQL-query.

```
SELECT ?term
    (count(?correct_answer) AS ?correct_answer_count)
    (count(?answer) AS ?answer_count)
    ((2*?correct_answer_count - ?answer_count)
        AS ?rank)
WHERE{
    ?module learningRu:hasTest ?test   .
    ?test ifmotest:hasGroupOfTasks
        ?group_of_tasks .
    ?group_of_tasks ifmotest:hasTask ?task .
    ?test_element lres:hasTask ?task .
    ?test_element lres:hasAnswer ?answer .
    ?task learningRu:hasTerm ?term .
    OPTIONAL {
        ?task ifmotest:hasCorrectAnswer
            ?correct_answer
        filter( ?correct_answer = ?answer)
    }
}
GROUP BY ?term
ORDER BY ASC(?rank)
```

[9] https://www.python.org/

[10] https://www.djangoproject.com/

[11] http://rdflib.github.io/sparqlwrapper/

The analysis of troublesome terms in tests is performed inside the module. Each module has a separate analytics page that contains widgets, tables and carts. The analytics page is the wiki page. The wiki page is based on the Semantic MediaWiki syntax[16] and stored inside the Information Workbench system. The data of all UI elements on the page are obtained by using SPARQL queries. The system analytics page of the module includes a bar char of the five most difficult terms for students and a table of all terms in the module with the knowledge rating.

The user interface of the troublesome terms statistics is shown in fig. 4.

9 Conclusion

The developed modules for the e-learning system provide teachers with a tool to maintain relevance and high quality of existing knowledge assessment modules. The system provides rating of the terms, which caused difficulties for students. Based on this rating, teachers can change theoretical material of the course by improving description of certain terms and add proper tasks.

The front-end of the e-learning system can be found at `http://openedu.ifmo.ru`.

Example of subject terms 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. 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. 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. The term knowledge rating can be also refined after its replacement by the proper ranking formula.

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. 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)
4. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z. Dbpedia: A nucleus for a web of open data (pp. 722-735). Springer Berlin Heidelberg. (2007)
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. Chakkrit Snae and Michael Brueckner Ontology-Driven E-Learning System Based on Roles and Activities for Thai Learning Environment. Interdisciplinary Journal of Knowledge and Learning Objects (2007)
10. 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)
11. Silberztein M. NooJ for NLP: a linguistic development environment. URL: http://www.NooJ4nlp.net/pages/NooJ.html (2002 - :)
12. Silberztein M. NooJManual [Electronic resource]. P.99. URL: http://www.NooJ4nlp.net/NooJManual.pdf, (2003)
13. Ibid. P.82
14. Holovaty, A., Kaplan-Moss, J. The Definitive Guide to Django. Estados Unidos: Editorial Apress, 72-73 (2009)
15. Gearon, P., Passant, A. Polleres, A. SPARQL 1.1 Update. World Wide Web Consortium, (2013)
16. Krötzsch, M., Vrandečić, D., Völkel, M. Semantic mediawiki. The Semantic Web-ISWC 2006. Springer Berlin Heidelberg, 935-942 (2006)

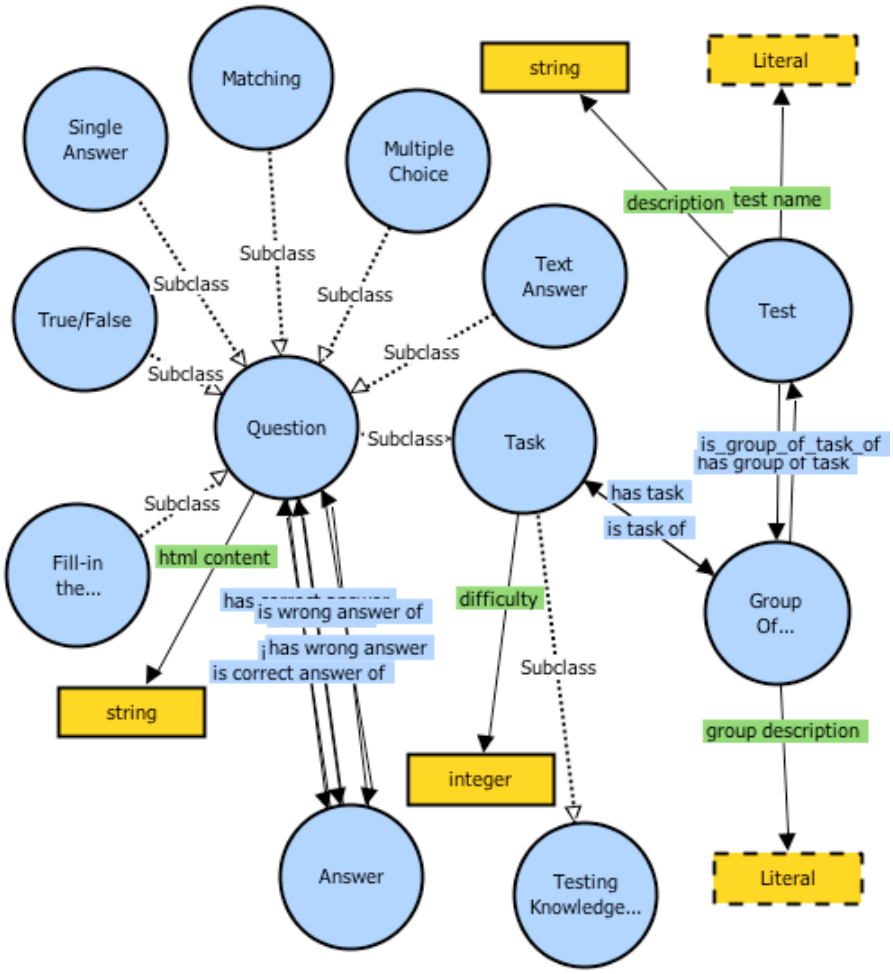


Fig. 1. Main classes of the test ontology

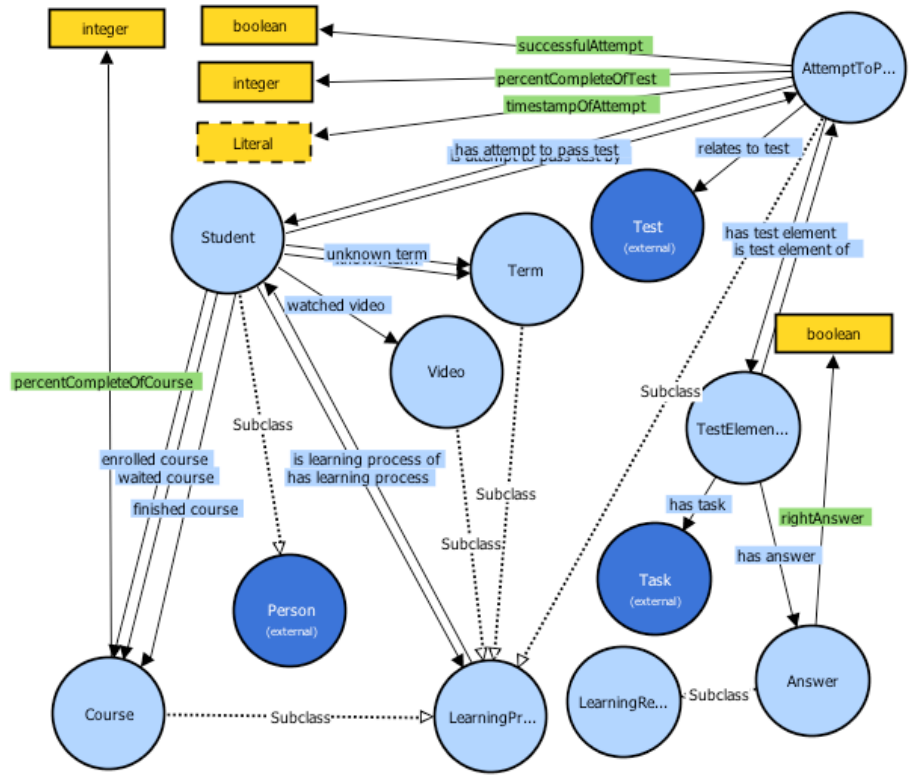


Fig. 2. Main classes of the student activity in the e-learning system ontology

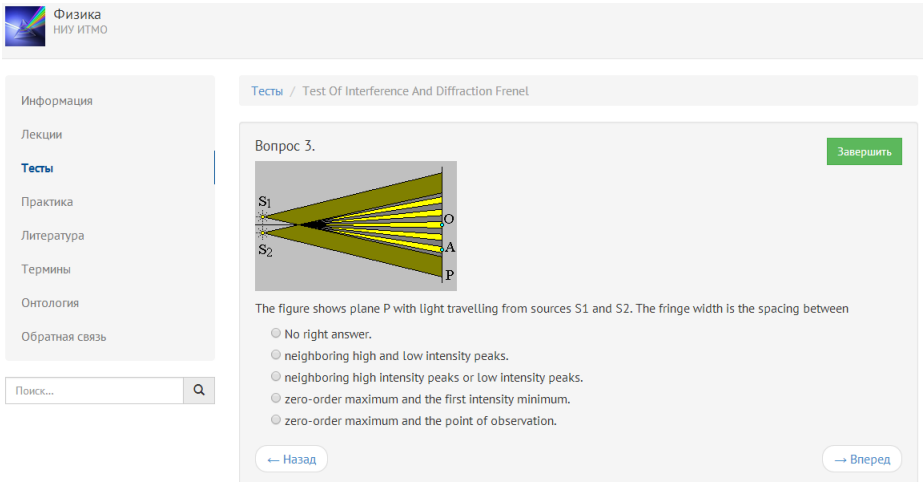
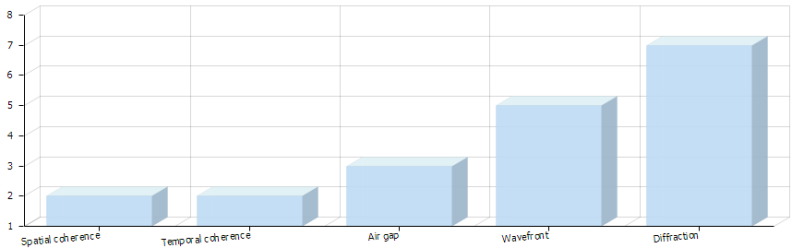


Fig. 3. The user interface of the front-end test page

Unclear Terms



Filter

Term	Correct Answers	All Answers	Rank
Spatial coherence	6	10	2
Temporal coherence	6	10	2
Air gap	4	5	3
Wavefront	7	9	5
Diffraction	9	11	7
Intensity	11	14	8
Wavelength	18	27	9
Fringes	20	29	11
Path	24	32	16
The interference pattern	30	42	18
Interference	39	54	24

Fig. 4. The user interface of the troublesome terms statistics