Platform for Computer-aided Harmonization of Informatics Curricula

Milinko Mandić[1], Zora Konjović[2], Mirjana Ivanović[3]

[1] Faculty of Education, University of Novi Sad, Podgorička 4, 25000 Sombor, Serbia, milinko.mandic@pef.uns.ac.rs

[2] University of Singidunum, 32 Danijelova St., 11 000 Belgrade, Serbia, zkonjovic@singidunum.ac.rs

[3] Faculty of Science, University of Novi Sad, Trg DositejaObradovića 3, 21000 Novi Sad, Serbia, mira@dmi.uns.ac.rs

Abstract: This paper presents a new platform aimed at improving informatics teaching by computer-aided harmonization of the standardized secondary school informatics curriculum and curricula by which teachers of informatics are educated. The platform relies on competency based curricula ontologies and the harmonization method based on ontology alignment. The secondary school informatics curriculum ontology was built to comply with the ACM K12 standard, while the teachers' curriculum ontology was built based on selected existing curricula, due to the lack of explicit standardization in the field. A task-specific method for curricula harmonization was developed that relies on standard ontology alignment algorithms. The prototype software tool was implemented and used by independent experts to verify the proposed method, by investigating compliance of the standardized secondary school informatics curriculum and the domain (informatics) segment of the teachers' curriculum.

Keywords: Informatics education; curriculum; ontology alignment; ACM K12

1 Introduction

The research presented in this paper was motivated by well recognized needs for frequent and even substantial changes in informatics teaching curricula at primary and secondary education levels caused by the extreme dynamics of changes in the informatics field and its complexity, along with labor market increasing IT competences requirements regarding all professions and all qualification levels. This gives an important role to existing IT competences and shifts the educational paradigm "from an input-centered approach to an output-focused student-centered approach" [1]. In order to keep pace, curricula for educating informatics teachers must be changed to respond by ensuring the necessary teachers' competences.

Hence, the representation of informatics curricula for educating informatics teachers and informatics curricula of lower levels of education is needed as well as tools that will facilitate curricula changes while keeping them compliant in terms of required informatics teachers' competences.

The rest of the paper is organized as follows. The second section presents related work. Section three presents briefly, the proposed ontological models of the curriculum for educating informatics teachers and the informatics curriculum for secondary education level. Section four presents the procedure underlying the software tool for curricula harmonization. The fifth section presents verification of the proposed platform by means of investigation of the compliance of the standardized secondary school level informatics curriculum and the domain (informatics) segment of the proposed teachers' curriculum. Finally, the sixth section contains concluding remarks, which include an evaluation of the achieved results and directions for further research.

2 Related Work

In accordance with the research presented in this paper (informatics curricula harmonization by ontology matching with an emphasis on acquired competences), the papers dealing with the application of ontology for the representation of the curricula and papers dealing with ontology matching and its applications to curricula harmonization were analyzed.

Ontological approaches are increasingly being applied to represent curricula, since ontology is machine-readable, reusable and sharable [2] [3] [4] [5]. Ontologies can represent the educational domain from different perspectives [6] [7], providing "a richer description and retrieval of learning contents" [2]. According to [3], ontologies are most appropriate for the development of curricula based on intended learning outcomes, students' competence and standards. In [4], a proposal for an ontology curriculum in the field of computing is provided and an idea of applying ontologies is described by which the user can choose from a drop down menu the desired learning outcome and, in accordance with the selected outcome, the corresponding concepts in the ontology developed are labeled. In [2], ontologies are applied as a basis of software for the development and maintenance of an educational curriculum that provides information on the length of instructional units, the duration of instruction, assessment instruments and the display of untaught lessons and the like. Demartini et al. [5] present an ontology representing the academic environment as suggested by the Bologna reform. The proposed ontology does not contain an explicit representation of the curriculum. Gluga et al. [8] describe a system that models curriculum design in university teaching programs. The system exploits a lightweight semantic mapping approach to map learning goals from multiple accrediting sources across the degree. In [9], a system for representing ACM CS curriculum based on the IEEE RCD standard is shown.

A range of different techniques and strategies for ontology alignment have been implemented in a number of systems, as is evident in [10] [11] [12]. Despite wide use of ontologies' application for representing curricula, as well as numerous publications dealing with researches of matching and alignment of ontologies such as [13] [14], in contemporary literature one can rarely find examples of implemented systems for the alignment of ontological representations of the curricula (different or the same levels of education). In [15] the authors emphasize the importance of a system for harmonizing curricula that have been modeled using ontologies. Conceptual maps were created describing the curricula translated into an ontology, where algorithms for alignment of study programs were neither described nor implemented.

3 The Ontological Model of Curricula

The main goal of the research presented in this paper was to propose a tool that would help in determining whether teacher education curriculum provides the competencies required for teaching in a high school. Therefore, the models of teacher education and secondary school informatics curricula are based on competencies and as such, the base class of both ontological models is *Competence*. Numerous definitions of competence [16] [17] [18] all agree with what is presented in [19], i.e., that the notion of competence, regardless the context, refers to successfully performing a task or activity, that is adequate acquaintance of some domain's ***knowledge*** or ***skill***. Therefore, in this paper, the knowledge and skills mapped to specific classes of an ontological model curriculum (*Knowledge* and *Skills*), are represented as subclasses of *Competence* as described in detail in [20]. Thematic areas of the curriculum are mapped to subclasses of the *Knowledge* class, whereas the skills acquired through the study of specific subject areas are mapped to the corresponding subclasses of the *Skills* class. The *Skills* subclasses and the *Knowledge* subclasses are related via the object property *hasKnowledge*, that is its inverse property *hasSkill*. To ensure interoperability with learning management systems that provide information about competence, upper ontology classes are modeled in accordance with the IEEE RCD standard as described in [9].

Analysis of the content and form of teacher education curricula available on the web sites of institutions in several countries (Germany, Austria, Turkey and the Republic of Serbia) shows that competencies corresponding to each subject (course) are determined primarily by two fields: *course content* and *course outcome*. In our model of curriculum *course content* corresponds to the *Knowledge* class and *course outcome* to the *Skills* class. Skills are represented by classes corresponding to the categories of the cognitive process dimension of the revised Bloom's taxonomy [21], which is the dominant taxonomy in the area of CS and in general [22]. Exceptions are 'remember' and 'understand' categories,

which are represented by a single class *Remember-understand*. Thus, the *Skills* subclasses are: *Remember-understand*, *Apply*, *Analyze*, *Evaluate* and *Create.*

Since no proposals of standardized curricula models for informatics teachers' education exists yet, an ontological model of a teacher education curriculum was created based on our analysis of 22 teacher education curricula from different countries (Germany, Austria, Israel, Estonia, Turkey, Scotland, USA and R. of Serbia), as well as the recommendations suggested by [23] [24]. Five general areas that all curricula for informatics teacher preparation must include are: Informatics (domain) knowledge, General pedagogical knowledge (educational psychology, didactics, etc.), Knowledge of the methods of teaching informatics, Knowledge of teaching practice, General knowledge (foreign languages, mathematics, the application of ICT in the realization of teaching). These five general areas were modeled by subclasses of the class *Knowledge*.

The hierarchical structure of the upper subclasses of the *Informatics_domain_knowledge* class is based on the classifications shown in [4] [25] [26]. The ontological model includes all areas of informatics knowledge contained in most of the analyzed curricula. In the ontological model of the teacher education curriculum descriptions of classes were further mapped to labels. Subclasses of the *Skills* class were created primarily based on ISTE standards specified in [24] [27]. *Skills* subclasses were also based on the outcomes/objectives of the courses contained in the analyzed teacher education curricula. Based on [21] [28], all the described teaching skills were classified in the appropriate subclasses of Bloom's taxonomy classes and then associated with the knowledge to which they can relate.

The ontological model of secondary school informatics curriculum in this paper was designed strictly following competences designed for the secondary level of education (K8 or higher levels of standard) of the ACM K12 CS curriculum proposal [29]. The ontological model of the secondary school informatics curricula is created in two phase as described in detail in [20].

Using the tool Protégé (http://protege.stanford.edu/), OWL ontologies representing high school and teachers' informatics curricula are created, which are available at addresses www.pef.uns.ac.rs/SecondaryInformaticsCurriculum/index.html and www.pef.uns.ac.rs/InformaticsTeacherEducationCurriculum/index.html respectively.

4 Method for Curricula Harmonization

For two ontologies O_1 and O_2, matching implies the process of finding an appropriate entity from O_2 for each entity from O_1. Alignment of ontologies is the output of the matching process and comprises a set of "correspondences" [13] between ontologies.

Since the object and data type properties are predefined in advance and are the same in both ontologies modeling curricula, the proposed method for curricula harmonization compares only classes of ontologies, so the harmonization model can be formally written as follows.

If the ontologies modeling two curricula are $O_1$and O_2 , C_{ik} is an ontology class, (=), (⊇), (⊆) are equivalence, one-to-many superset/superclass and one-to-many subset/subclass relations respectively and conf_i is degree of confidence, then the curricula harmonization model is

$$\text{Alignment}(O_1, O_2) = \left\{ \left(C_{i1}, C_{j2}, \text{conf}_i, \text{relation}_i \right) \middle| \, C_{i1} \in O_1, C_{j2} \in O_2, \text{conf}_i \in [0,1], \text{relation}_i \in \{=, \subseteq, \supseteq\} \right\}.$$

Figure 1 shows the diagram of the method proposed in this paper for matching the secondary school and teacher education curricula.

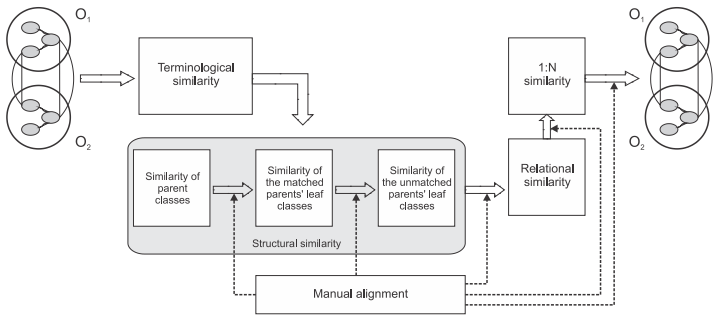

Figure 1
The procedure of matching secondary school and teacher education curricula

The matching is done in two phases. The first phase, which can be considered as pre-processing, determines terminological similarity by means of linguistic and string-based method [13] applied to local names and the classes' labels. The obtained similarity matrix is input to the second phase, which consists of the sequential composition of matchers determining structural, relational and one-to-many similarities respectively. Each matcher of this phase provides input (similarity matrix) to the subsequent matcher. The best matched pairs of classes are determined by applying the greedy selection algorithm as described in [30]. Three matchers that calculate structural similarity compare only subclasses of the *Knowledge* class in teachers' curriculum to which topics from domain (informatics) knowledge are mapped with subclasses of *Knowledge* class of secondary school curriculum because classes that belong to non-informatics knowledge in teacher education curriculum appear in teachers' curriculum only. Matching of skills structures (subclasses of the *Skills* class) is determined through relational similarity with an aim to check whether the secondary school skills are at the lower or the same level of Bloom's taxonomy with matched teaching skills.

User manual intervention is enabled after each matching stage, except after the terminological stage. Following manual interventions are enabled that preserve the consistency of one-to-many relations cardinality (e.g. superset/superclass and subset/subclass) produced by the matcher:

a) Changes in the greedy algorithm's threshold values
b) Disconnection of the matched classes
c) Changing the correspondence degree of matched classes
d) Replacement of the class in the matched pair
e) Creating a new matched pair of classes

The rest of this section contains descriptions of applied alignment algorithms and rationales for the choice of algorithms.

4.1 Terminological Similarity

Terminological similarity is determined by applying standard linguistic method based on the WordNet lexical database to strings that identify particular class. Labels are used for the additional description of concepts in the curricula; thus, when comparing classes of two ontologies using a terminological matcher, local class names and their labels are taken into account.

The similarity between two tokens belonging to the local names of classes is determined using the Lin information-theoretic similarities [31] in instances where there are two tokens in the WordNet dictionary. If this is not the case, token similarity is determined using the Jaro-Winkler method [32] [33]. Applying the greedy selection method to a matrix consisting of the similarities of all possible pairs of tokens of compared names of classes, a list S_{ln} is obtained that contains similarities of the best matched pairs of tokens. The total similarity of local names for the two classes $s_{ln}(C_{i1},\ C_{j2})$ is calculated as:

$$s_{ln}(C_{i1}, C_{j2}) = \frac{2 \cdot \sum_{i=o}^{m} S_{ln}(i)}{|tok_{i1}| + |tok_{j2}|}\ ;\ |tok_{ik}| -\ \#\ \text{of tokens in local name of } C_{ik};\ \ m - \text{dimension of } S_{ln}$$

The similarity of classes' labels $s_{lb}(C_{i1},\ C_{j2})$ and the similarities between the local name of the class of one ontology and the label of the class of other ontology ($s_{lnlb}(C_{i1},\ C_{j2})$ and $s_{lbln}(C_{i1},\ C_{j2})$) are calculated analogously. The total terminological similarity for classes $s_{term}(C_{i1}, C_{j2})$ is:

$$s_{term}(C_{i1}, C_{j2}) = \max(s_{ln}(C_{i1}, C_{j2}), s_{lb}(C_{i1}, C_{j2}), s_{lnlb}(C_{i1}, C_{j2}), s_{lbln}(C_{i1}, C_{j2}))$$

4.2 Structural (taxonomic) Similarity

Structural (taxonomic) similarity is calculated in three steps:

- Calculating the similarities of all parent classes

- Calculating the similarities of non-parent (leaf) classes that are subclasses of matched parent classes
- Calculating the similarities of non-parent classes that are subclasses of unmatched parent classes

Such composition of structural algorithms enables manual intervention in order to support early correction which is necessary because the results of subsequent matchers depend on the results of the previous ones.

4.2.1 Determining the Similarity of Parent Classes

For the similarity of parent classes a slight modification of the algorithm presented in [11] is used.

For two parental classes C_{i1} and C_{j2}, the similarities of their superclasses ("parents"), the similarities of their subclasses ("children") and their terminological similarity are taken into account. There are observed similarities of *all parents and children*, not only of direct ones. Similarity between the subclasses of C_{i1} and C_{j2}, denoted by $s^{sub}(C_{i1}, C_{j2})$, is determined by the following algorithm:

/* Let A_{ij} be a class of an ontology, $A_{i1} \in O_1$ and $A_{i2} \in O_2$

If $\nexists A_{i1} | A_{i1} \subseteq C_{i1}$ or $\nexists A_{i2} | A_{i2} \subseteq C_{j2}$ then

$s^{sub}(C_{i1}, C_{j2}) = 0$

else

Let $\{A_{k1}\} \subseteq C_{i1}, k = 1, n;\ n \geq 1$ and $\{A_{l2}\} \subseteq C_{j2}, l = 1, m;\ m \geq 1$

for k = 1 to n

for l = 1 to m

/* $s_{term}(A_{k1}, A_{l2})$ are the values of similarity of classes from the set $\{A_{11} \dots A_{n1}\}$ with classes from the set $\{A_{12} \dots A_{m2}\}$

submatrix[k][l] = $s_{term}(A_{k1}, A_{l2})$

/* the list of best-matched pairs of subclasses S^{sub} is obtained applying the greedy selection method to the submatrix

S^{sub} = Greedy_Selection_Method (submatrix)

/* $s^{sub}(C_{i1}, C_{j2})$ is set to the average value of similarities of *matched* subclasses

$s^{sub}(C_{i1}, C_{j2}) = \frac{\sum_{l=o}^{p} S^{sub}(l)}{p}$, p= size of S^{sub}

The similarity of superclasses $s^{sup}(C_{i1}, C_{j2})$ is calculated analogously using similarities of each superclass of the C_{i1} class with each superclass of the C_{j2}class, and calculating the average of the matched superclasses. Overall similarity

$s_{parent}(C_{i1}, C_{j2})$, is calculated as the average of the terminological similarities and previously calculated similarities of superclasses and subclasses, provided that both classes C_{i1} and C_{j2} have at least one subclass; if the condition is not met overall similarity is 0. Modification of algorithm [11] takes place if one of compared classes has no parent class. In this case, value $s^{sup}(C_{i1},\ C_{j2})$ is omitted when calculating average for $s_{parent}(C_{i1}, C_{j2})$. That way the impact of the structural similarity is relaxed, leaving larger number of potentially useful classes for further matching which is reasonable taking into account the fact that teachers' and high school curricula have relatively different structures. The similarity matrix of this structural matcher S_{parent} has mxn dimension where *m* and *n* are the total number of *Knowledge* subclasses in ontologies O_1 and O_2, respectively. The list of matched classes A_{parent} is obtained by applying the greedy selection algorithm to the matrix S_{parent}. The similarities of predefined classes (*Knowledge, Competence*) are not taken into account in these calculations.

4.2.2 Determining the Similarities of the Matched Parents' Leaf Classes

At this stage, the similarity $s_{leaf}(C_{i1}, C_{j2})$is calculated as follows:

/* Let A_{ij} be the class of the ontology, $A_{i1} \in O_1$and $A_{j2} \in O_2$.

/* Further, let the following apply: C_{i1} is a leaf class of ontology $O_1$and C_{j2} is a leaf class of ontology O_2, or C_{i1} is a leaf class of ontology O_1 and the C_{j2} class has only leaf subclasses, or C_{j2} is a leaf class of ontology O_2 and C_{i1} has only leaf subclasses.

If $\exists\{A_{l1}, A_{k2}\} | \{A_{l1}, A_{k2}\} \in A_{parent}, A_{l1} \in \{A_{11} \dots A_{n1}\}, C_{i1} \subseteq \{A_{11} \dots A_{n1}\}, A_{k2} \in \{A_{12} \dots A_{m2}\}, C_{j2} \subseteq \{A_{12} \dots A_{m2}\}$then

$$s_{leaf}(C_{i1}, C_{j2}) = s_{term}(C_{i1}, C_{j2})$$

else

$$s_{leaf}(C_{i1}, C_{j2}) = s_{parent}(C_{i1}, C_{j2})$$

In order to avoid elimination of potentially equivalent classes that are not described with the same level of detail (by subclasses), in addition to the comparison of leaf classes, the comparison of non-leaf classes having only leaf subclasses with the leaf classes is also done.

4.2.3 Determining Similarities of the Unmatched Parents' Leaf Classes

The similarity of leaf classes C_{i1} and C_{j2} becomes zero, if no matching of the parents of C_{i1}, with any parent of C_{j2} is obtained by applying the first two structural matchers. This, together with curriculum description, which is far from being unambiguous for non-standardized curricula, could leave some essentially related concepts (with different parents), unpaired. For example, in the secondary school curriculum model the concepts of computer graphics are represented as subclasses of the *Multimedia* class, while in many teaching curricula, concepts

relating to computer graphics and those relating to multimedia, belong to distinct courses, i.e., in the teacher education curriculum, computer graphics concepts are represented as special subclasses of the *Graphics* class, while there is a separate parent class *Multimedia* containing no computer graphics concepts at all. Since the class *Multimedia* of the secondary school curriculum model is not matched with the *Graphics* class of the teacher education curriculum model, but with the *Multimedia* class, the previous matcher would calculate zero similarity measure between classes to which, for example, concepts of raster images are mapped. This problem is resolved here by explicitly defining disjointed parent classes, i.e., the classes *Multimedia* and *Graphics* of the teachers' curriculum are not defined as disjoint. Then, the principle for determining the similarity of classes $s_{disj}(C_{i1}, C_{j2})$ is as follows:

/* Let A_{leaf} be a list of matched classes obtained by a matcher that determines the similarity of leaf classes of matched parents.

/* Let the following apply:$\{\{A_{11}, A_{12}\} \dots \{A_{n1}, A_{n2}\}\} \in A_{leaf}$,$C_{i1} \subseteq \{A_{11} \dots A_{n1}\}$, $C_{j2} \subseteq \{B_{12} \dots B_{m2}\}$, $A_{k2} \in \{A_{12} \dots A_{n2}\}$,$B_{k2} \in \{B_{12} \dots B_{m2}\}$

If C_{i1} and C_{j2} are unmatched leaf classes and $\nexists A_{k2}$, B_{k2} defined as disjoint classes and $\nexists\{A_{l1}, B_{k2}\} \mid \{A_{l1}, B_{k2}\} \in A_{leaf}$, $A_{l1} \in \{A_{11} \dots A_{n1}\}$, $B_{k2} \in \{B_{12} \dots B_{m2}\}$ then

$$s_{disj}(C_{i1}, C_{j2}) = s_{term}(C_{i1}, C_{j2})$$

else

$$s_{disj}(C_{i1}, C_{j2}) = s_{leaf}(C_{i1}, C_{j2})$$

This matcher in the sequential composition is after the matcher determines the similarity of matched parents' leaf classes, which favors matched parents' classes, but also extends the search space to other non-disjoint classes that could contain some useful concepts.

All structural matchers calculate similarities only between the *Knowledge* subclasses, so the similarities of the subclasses of the *Skills* class are not changed by structural alignment step.

4.3 Determining Relational Similarity

The outcomes/objectives of the course or subject areas in our ontological models are simply mapped to the corresponding subclasses of Bloom's taxonomy classes, which are the subclasses of the *Skill* class. This makes determination of the similarity of classes on the basis of their taxonomic structure inappropriate for this part of the ontology. On the other hand, the outcomes of the curricula (mapped to the appropriate *Skills* subclasses in the ontology) are usually described by a larger free text, which makes the use of only a terminological matcher inappropriate. Therefore, the similarity of *Skills* subclasses in the system is calculated based on the relation graph. The method for calculating relational similarity applied in the

paper is based on the principle used in [34]: *if the two classes that represent the domains of object properties (relation) are similar, and if the object properties are also similar, then the classes representing the ranges of the domain classes are similar* [13]. Relational similarity $s_{rel}(C_{i1}, C_{j2})$ is determined as follows:

/*Let A_{disj} be a list of matched classes obtained by a matcher that determines the similarity of leaf classes of unmatched parents

/* Let $C_{Knowledgei}$ be the *Knowledge* class and let $C_{Skillsi}$ be the *Skills* class

If $C_{i1} \subseteq C_{Knowledge1}$ or $C_{j2} \subseteq C_{Knowledge2}$ then

$$s_{rel}(C_{i1}, C_{j2}) = s_{disj}(C_{i1}, C_{j2})$$

else if $C_{i1} \subseteq C_{Skills1}$ and $C_{j2} \subseteq C_{Skills2}$ then

If C_{i1} is associated with $\{A_{11} \dots A_{n1}\} | \{A_{11} \dots A_{n1}\} \subseteq C_{Knowledge1}$ and C_{j2} is associated with $\{A_{12} \dots A_{m2}\} | \{A_{12} \dots A_{m2}\} \subseteq C_{Knowledge2}$ then

If $\{A_{k1} \dots A_{l1}\}$ is the set of all superclasses and subclasses of all classes from $\{A_{11} \dots A_{n1}\}$, $k = n + 1$ and $\{A_{o2} \dots A_{p2}\}$ is the set of all superclasses and subclasses of all classes from $\{A_{12} \dots A_{m2}\}$, $o = m + 1$ then

If $\exists\{A_{q1}, A_{r2}\} | \{A_{q1}, A_{r2}\} \in \mathrm{A_{disj}}$, $A_{q1} \in \{A_{11} \dots A_{n1}\} \cup \{A_{k1} \dots A_{l1}\}$, $A_{r2} \in \{A_{12} \dots A_{m2}\} \cup \{A_{o2} \dots A_{p2}\}$ then

$$s_{rel}(C_{i1}, C_{j2}) = s_{term}(C_{i1}, C_{j2})$$

else

$$s_{rel}(C_{i1}, C_{j2}) = 0$$

If a structure exists in the part of the ontology to which the subclasses of the *Skills* class belong (some outcomes are further structured), then for these subclasses, when calculating a relational similarity, the relations inherited from their superclasses are taken into consideration. Due to the fact that in our model, the object property that connects *Knowledge* and *Skills* subclasses is known and the same in both ontologies, "the circularity" which could be caused by using the relational method [13] is reduced (the similarity of object properties based on the similarity of the domain and range is not explicitly calculated).

4.4 Determining 1:N Similarity

Previously described algorithms determine to what extent the classes of ontology O_1 are *equivalent* to the classes of ontology O_2 with cardinality of 1:1. The next alignment phase enables matching of a class of one ontology with *multiple classes* of the other ontology through relation *superclass/subclass*. The following pseudo-code describes the method that determines whether some class C_{i1} from O_1 is a superclass of classes from O_2.

/* Let A_{rel} be a list of matched classes obtained by a matcher determining the relational similarity

If $\{C_{i1}, C_{j2}\} \in A_{rel}$ and $\nexists A_{l1} | A_{l1} \subseteq C_{i1}$ and $\exists A_{k2} | A_{k2} \subseteq C_{j2}$ then

If $\nexists \{A_{l1}, A_{k2}\} | \{A_{l1}, A_{k2}\} \in A_{rel}, A_{l1} \in O_1,\ A_{k2} \in \{A_{12} \dots A_{n2}\},\ \{A_{12} \dots A_{n2}\} \subseteq C_{j2},\ n \geq 1$ then

$$\{A_{12} \dots A_{n2}\} \subseteq C_{i1}$$

An analogous procedure is applied to determine whether the class C_{j2} is a superclass of classes from O_1. After applying this method, a class can be associated with several classes of the other ontology by superclass and equivalence relations. Conversely, a class can be a subclass of the ontology class to which it belongs, as well as, the class of the other ontology.

5 Verification of the Proposed Curricula Harmonization Method

Based on the models and algorithms described in Sections 3 and 4 of this paper, the software application for curricula harmonization was implemented using the Java programming language. Evaluation of the software was carried out by the expert team composed of 4 university professors in the field of informatics teacher education, 2 employees in the Education District Offices (Ministry of Education) and 2 teachers teaching secondary school informatics. Their tasks were to define the reference alignment and to interpret the results. In the rest of this section the results obtained by the software tool application to the curricula from Section 3 and the experts' analysis of these results are presented following the matching steps (matchers) applied after terminological matching.

5.1 Similarity of the Parent Classes

Figure 2 shows a part of the matched classes of compared curricula obtained by the first taxonomical/structural algorithm that determines the similarity of classes that have at least one subclass, with the threshold set to 70%.The percentage of matched *Knowledge* subclasses at this stage was 14.9%.

The column "Source class" and "Target class" contain the local names of classes of ontological representations of secondary school and teacher education curricula, respectively; the column "Type of relation" identifies the type of relation between the classes (Equivalence, Superclass and Subclass), while "Similarity Value" denotes the correspondence value between the matched classes.

The expert team noticed that certain classes with identical names were matched with the similarity value below 100% and that some classes were matched despite

not having similar names (Figure 2). The explanation for this is the presence of an additional description in the labels of teacher education curriculum for some classes and/or the participation of similarities of superclasses/subclasses in the calculation of the overall similarity of classes.

Row	Source class	Target class	Type of relation	Similarity...
1	Algorithms	Algorithms	Equivalence	83.18%
2	Connections_Between_Mathematics_and_Comp...	Mathematical_basis_of_informatics	Equivalence	76.35%
3	Data_Structures	Data_types_and_structures	Equivalence	91.31%
4	Databases	Database	Equivalence	89.32%
5	Fundamentals_of_Hardware_Design	Memory	Equivalence	75.33%
6	Hypertext_Language_HTML_tags	HyperText_Markup_language_-_HTML	Equivalence	86.7%
7	Levels_of_Language_Software_and_Translation	Programming_paradigms	Equivalence	70.02%
8	Models_of_Intelligent_Behavior	Artificial_intelligence	Equivalence	74.31%
9	Multimedia	Multimedia	Equivalence	85.42%
10	Object-oriented_programming	Object-oriented_programming	Equivalence	94.52%
11	Parts_of_a_Computer	Hardware_basics	Equivalence	75.07%
12	Phases_of_the_software_development_process	Models_and_phases_of_the_software_deve...	Equivalence	89.52%
13	Principles_of_Software_Engineering	Software_engineering	Equivalence	90.78%
14	Principles_of_computer_organization	Architecture_and_Organization	Equivalence	79.44%
15	Problem_Solving_and_Algorithms	Problem_solving	Equivalence	89.84%
16	Problem_solving	Problem_solving_phases	Equivalence	84.13%
17	Programming_Languages	Programming_Fundamentals	Equivalence	92.03%
18	Representing_Information_Digitally	Data_representation	Equivalence	77.02%
19	Structured_programming	Structured_and_Imperative_programming	Equivalence	87.57%
20	The_major_component_parts_of_the_microproc...	Central_processing_unit_-_CPU	Equivalence	77.29%
21	Web_Page_Design_and_Development	Web_technologies_and_development	Equivalence	76.76%

Figure 2
Matched classes after applying the algorithm for parent classes matching

In addition, it was found that some classes having the same names in the secondary school curriculum and teacher education curriculum (for example, *Problem solving*) were not mutually matched, but that the *Problem_solving* class of the secondary school curriculum and the *Problem_solving_phases* class in the teacher education curriculum were matched (row 16); the expert team considered this as correct, because the subclasses of both matched classes represent stages in algorithmic problem solving.

Additionally, looking only at the names of the matched classes from Figure 2, the matching of the classes *Levels_of_Language_Software_and_Translation* and *Programming_paradigms* (row 7) could be considered as false. However, the topics of secondary school and teacher education curricula (differences and comparison of high level languages and machine languages, levels of programming languages, etc.) described by their subclasses are corresponding.

At this level of the application of a structural matcher, the expert team identified a pair of incorrectly matched classes *{Fundamentals_of_Hardware_Design, Memory} (*row *5).* However, since their parent classes were correctly matched, this pair of classes does not influence the similarity of their subclasses, which will be calculated by the following matchers.

5.2 Similarities of the Matched Parents' Leaf Classes

Figure 3 displays some matched classes obtained after applying a taxonomic/structural algorithm that determines the similarity between leaf classes of the matched parents. The percentage of matched *Knowledge* subclasses at this stage was 61.18%.

The similarity of the matched classes obtained at this stage was determined by the terminological similarity of their local names and labels, under condition that some of their parent classes were matched by the matcher calculating the similarity of parent classes, which explains why classes *Repetition* and *Iteration* were highly matched (Figure 3, row 17). Namely, their non-direct parent classes *Programming_Languages* and *Programming_Fundamentals* had already been matched (Figure 2, row 17). Further, since the verbs *repeat* and *iterate* are considered as synonymous within the WordNet database, the terminological matcher showed high similarity for the *Repetition* and *Iteration* classes.

An example of matching a leaf class to a class that is the parent of leaf classes is the match {*Knowledge-based_Systems, Semantic_Web_and_knowledge_ representation*} (Figure 3, row 12). The class *Knowledge-based_Systems* has no subclasses and is a subclass of the *Models_of_Intelligent_Behavior* class. The class *Semantic_Web_and_knowledge_representation* has subclasses and is a subclass of the *Artificial_intelligence* class matched with the class *Models_of_Intelligent_Behavior* by applying the matcher for calculating the similarities of parent classes (Figure 2, row 8).

R...	Source class	Target class	Type ...	Similar..
1	Careers_related_to_computers	Profession_and_Careers_in_computing	Equiv...	83.15%
2	Challenges_of_modeling_information_digitally	Representation_of_the_different_types_of_informati...	Equiv...	71.78%
3	Client_side_scripts_in_a_networked_environment	Client-side_scripting	Equiv...	75.0%
4	Code_a_solution_from_a_design	Software_deployment	Equiv...	80.0%
5	Conversion_among_decimal,_binary,_and_hex_number_sy...	Conversion_among_different_number_systems	Equiv...	93.33%
6	Creating_a_web_site_that_conforms_to_standards	Basic_Principles_of_creating_web_sites	Equiv...	77.46%
7	Diagnose_and_troubleshoot_PC_problems	Maintenance_and_support_of_PC_hardware	Equiv...	70.02%
8	Encoded_data_and_integrated_circuits	Characteristics_of_digital_integrated_circuits	Equiv...	77.44%
9	Hardware_to_support_multimedia	Hardware_supporting_multimedia	Equiv...	100.0%
10	Interactivity	Static_and_Dynamic_web_content	Equiv...	76.89%
11	Interface_evaluation	Measures_for_evaluation_in_HCI	Equiv...	77.44%
12	Knowledge-based_Systems	Semantic_Web_and_knowledge_representation	Equiv...	80.16%
13	Natural_Language	Natural_language_processing	Equiv...	80.0%
14	Presentation_software	Software_applications_for_presentations	Equiv...	80.0%
15	Relationships_among_high-level_languages_assembly_l...	Higher_level_languages_vs_machine_level_langua...	Equiv...	73.35%
16	Relevancy_of_web_sources	Sharing_documents_on_the_web	Equiv...	71.0%
17	Repetition	Iteration	Equiv...	100.0%
18	Routing_protocols_for_connection-communication	Principles_of_routing	Equiv...	85.93%
19	Using_the_clipboard	Functions_of_interrupts	Equiv...	72.22%
20	What_is_Intelligence	Definition_of_artificial_intelligence	Equiv...	85.71%

Figure 3

Example of matched classes after applying the second structural algorithm

At this stage, the expert team reported substantially incorrect matches (row 16, 19), which were true candidates for manual interventions.

5.3 Similarities of the Unmatched Parents' Leaf Classes

According to the previous matcher, some subclasses of the *Multimedia* class *(Create_edit_and_save_bitmapped_images, Vector_versus_bit-mapped_images, Create_edit_and_save_vector_images)* of the secondary school curriculum had not been matched with subclasses of the *Multimedia* class of the teacher education curriculum. By applying the algorithm for calculating the similarities of the leaf classes of unmatched parents, these classes were matched with the subclasses of the *Graphics* class (Figure 4).The percentage of matched *Knowledge* subclasses at this stage was 82.35%.

...	Source class	Target class	Type of rela...	Simila...
1	Create_edit_and_save_bitmapped_images	Programs_to_create_and_edit_raster_graphics	Equivalence	79.5%
2	Create_edit_and_save_vector_images	Programs_to_create_and_edit_vector_graphics	Equivalence	81.49%
3	Vector_versus_bit-mapped_images	Methods_of_presenting_static_images_in_com...	Equivalence	100.0%

Figure 4

Matched leaf classes whose parents were not paired

Classes that remain unmatched after the application of the structural algorithm may indicate incompleteness of knowledge in the teacher education curriculum or incompatible structures of curricula ontologies. Examples of incompleteness in teacher education curriculum correctly detected by the system are machine cycle phases, robotics, documentation techniques and elements of user friendly software. Example of false incompleteness detected in the teacher curriculum, which is caused by incompatible structures of the curricula ontologies, are those related to connections between mathematics and computer science where the unmatched class *Functions_including_parameters_and_mathematical_notation* in the secondary school curriculum is a subclass of the class *Connections_between_mathematics_and_computer_science*, while in the teacher education curriculum corresponding knowledge was mapped to a subclass of the *General_knowledge* class that does not belong to the CS domain knowledge at all. Finally, differences in the structure of ontologies arising from the depth of studying specific topics in the secondary school and teacher education curricula may result in unmatched classes that do not necessarily point to an inadequate teacher education curriculum. An example is the thematic area of the secondary school curriculum 'Interdisciplinary Utility of Computers and Problem Solving in the Modern World' with focuses representing the various applications of computers including 'Education and Training'. Since these focuses were mapped to the leaf subclasses of the class *Interdisciplinary_utility_of_computers_and_problem_solving_in_the_modern_world* in the secondary school curriculum, despite the fact that the teacher education curriculum contains classes (such as *Educational_software* and *E-learning*) that correspond to the focus 'Education and training*'* from the secondary school curriculum, these classes were not matched with the leaf class *Education_and_training*, due to the fact that in the teacher education curriculum they have class structures not considered by the proposed matchers.

5.4 Relational Similarity

In terms of the lowly-structured subclasses of the *Skills* class (practically the only structure by which Bloom's taxonomy is modeled), where the titles and labels of subclasses usually contain free text, terminological matching significantly affects the final results. To avoid omitting potentially useful matches that can be used for manual intervention, in this instance, a lower criterion (threshold) was set in the determination of the matched classes (60%). Percentage of paired classes was 80.88%.

A part of the results obtained using the relational matcher determining the similarity between the subclasses of the *Skills* class is shown in Figure 5. The "Bloom" column in the table in Figure 5contains the ⊤ mark if the level of skill in the teacher education curriculum is higher or equal to the level required in the secondary school curriculum, or the ⊥ mark if not.

The opinion of the expert team was that some matched classes here are potentially inaccurate (rows 3, 7, 11 and 14). The classes that were not matched because there was no corresponding class in the teacher education curriculum were the classes *Explain_the_relationship_between_a_web_server_a_web_page_and_a_browser* and *Describe_the_difference_in_the_processing_of_arrays_stacks_and_queues*.

R...	Source class	Target class	Ty...	Simil...	Bloom
1	Convert_a_word_problem_into_code_using_top-down_design	Design_programs_in_languages_from_two_different_programming_par...	Eq...	71.45%	⊤
2	Convert_between_decimal_binary_and_hexadecimal_numbers	Apply_arithmetic_operations_in_different_number_systems	Eq...	70.04%	⊤
3	Convert_between_image_formats	Contrast_vector_and_raster_graphics	Eq...	77.24%	⊤
4	Create_a_Web_site_given_design_specifications	Design_web_pages	Eq...	69.16%	⊤
5	Create_a_user-centered_design	Design_interactive_user_interfaces_for_diverse_applications	Eq...	74.59%	⊤
6	Define_intellectual_property_and_state_the_impact_of_provisions_to_prote...	Discuss_intellectual_property	Eq...	64.16%	⊤
7	Define_parallel_processing	Use_design_patterns	Eq...	61.94%	⊤
8	Describe_the_major_applications_of_artificial_intelligence_and_robotics.	Apply_Artificial_intelligence_applications	Eq...	70.7%	⊤
9	Describe_the_role_of_the_OS_as_an_intermediary_between_application_...	Explain_the_objectives_and_functions_of_modern_operating_systems	Eq...	64.59%	⊤
10	Design_a_multi-table_relational_database	Project_relational_data_model	Eq...	75.0%	⊤
11	Determine_if_a_given_algorithm_successfully_solves_a_stated_problem	Select_basic_language_instructions_to_accomplish_a_given_straightfor...	Eq...	64.34%	⊥
12	Display_a_multimedia_object_within_a_Web_page_or_document	Set_the_multimedia_on_the_web	Eq...	61.21%	⊤
13	Evaluate_algorithms_by_their_efficiency_correctness_and_clarity	Analyze_algorithms_using_complexity_efficiency_aesthetics_and_correct...	Eq...	69.93%	⊥
14	Evaluate_computer_components_in_terms_of_features_and_price	Understand_machine_level_components_and_related_issues_of_comp...	Eq...	65.63%	⊥
15	Express_the_design_of_a_Web_site_using_standard_tools	Use_web_design_tools	Eq...	72.73%	⊤
16	List_ways_to_increase_computer_performance	Propose_options_to_improve_computer_performance	Eq...	67.85%	⊤
17	Name_and_explain_the_steps_in_the_problem-solving_process	List_problem_solving_phases	Eq...	72.81%	⊤
18	Name_the_different_phases_of_the_software_development_process	Use_one_or_more_software_development_models	Eq...	69.61%	⊤
19	Use_modeling_and_simulation_to_represent_and_understand_natural_ph...	Use_Modeling_and_simulation_to_solve_real_world_problems	Eq...	74.53%	⊤
20	Utilize_advanced_OS_user_interface_elements_and_features	Use_interactive_graphic_OS	Eq...	62.91%	⊤
21	Write_conditional_statements_that_include_simple_and_complex_Boolean...	Create_complex_logical_expressions_using_Boolean_operators_and_f...	Eq...	75.93%	⊤

Figure 5

A part of matched skills of the secondary school and teacher education curricula

The expert also reported that some outcomes in the secondary school curriculum were represented by a larger number of skills subclasses than the corresponding outcomes in the teacher education curriculum. Consequently, some skills from the secondary school curriculum remain unpaired, even when the teacher education curriculum contains classes that include these skills (such as *Code_a_program_to_solve_a_stated_problem_using_variables_and_at_least_one_decision_or_loop* and *Use_advanced_search_engine_options_and_refine_searches_to_locate_information*).

5.5 1: N Similarity

An example that justifies application of the 1:N algorithm is the matching of the subclasses of the *Semantic_Web_and_knowledge_representation* class and the *Knowledge-based_Systems* class. Since the class *Semantic_Web_and_knowledge_representation* contained unmatched leaf subclasses and the *Knowledge-based_Systems* leaf class was matched with *Semantic_Web_and_knowledge_representation* (Fig 3, row 12), the system suggested the 1:N relation, i.e., that the subclasses of the *Semantic_web_and_knowledge_representation* class (*Ontology, Predicate_logic, Web_ontology_language,* etc.) could also be the subclasses of the *Knowledge-*

based_Systems class (Figure 6).The total percentage of matched *Knowledge* subclasses achieved after the last matching phase was 87%.

...	Source class	Target class	Type of rela...	Similari...
1	Knowledge-based_Systems	Knowledge_representation_in_educa...	Superclass	80.16%
2	Knowledge-based_Systems	Ontology	Superclass	80.16%
3	Knowledge-based_Systems	Predicate_logic	Superclass	80.16%
4	Knowledge-based_Systems	Proposotional_logic	Superclass	80.16%
5	Knowledge-based_Systems	Resource_Description_Framework_-...	Superclass	80.16%
6	Knowledge-based_Systems	Semantic_web_-_basic_notions	Superclass	80.16%
7	Knowledge-based_Systems	Web_ontology_language	Superclass	80.16%

Figure 6
Matched classes in "Superclass" relation

5.6 Prototype Performance and Usability

Performance measures *Precision* (0.64), *Recall* (0.76) and *F-measure* (0.695) were obtained using reference alignment derived by human experts and results obtained by matching system, which is in accordance with reference [35] that gives maximum importance to the recall measure when ontology alignment is a semi-automatic process.

The expert team evaluated these results as acceptable. They also found the tool useful "as it is" for improving concrete teacher education curriculum in order to meet the requirements of the ACM K12 curriculum. The acquired class pairs evaluated as incorrect justify the need for the semi-automatic method for curricula harmonization.

The obtained quantitative results about the percentage of matched classes and the preliminary evaluation imply that the model of the teacher education curriculum is satisfactorily harmonized with the ACM K12 model. Still, the experts reported that even preliminary results obtained by means of the software prototype correctly indicate some subject areas that are not covered by the model of teacher education curriculum (machine cycle phases, documentation techniques, robotics, user-friendly web design, Interface evaluation, etc.) and that the teacher education curriculum does not provide all the skills needed for teaching in accordance with the ACM K12 curriculum proposal. Therefore, it is necessary to improve the teacher education curriculum so that it represents the missing knowledge and skills. In addition, some of the unmatched classes indicate incompatible structures of the ontological models. Typical examples are 'Connections Between Mathematics and Computer science' and 'Interdisciplinary Utility of Computers'. Such information makes a system useful for improvement of structure of the teacher education curriculum model. Also, some *Skills* classes of the ACM K12 model remained unmatched even in the teacher education model: there is the *Skills* class that could be considered as their superclass. Consequently, it is necessary to improve the teacher education curriculum so that the skills related to programming and the use of Internet be described in more detail/with a greater number of classes.

Conclusions and Future Work

The focus of this paper is the task-specific semi-automated method which can assist in development and maintenance of the teacher education curricula as to provide teachers' competences required by changes in the high school informatics curricula.

OWL ontologies of standardized secondary school informatics curriculum and the curriculum for the education of informatics teachers were developed, where the ontology of the secondary school curriculum relies on the ACM K12 standard, while the ontology of the teacher education curriculum was designed on the basis of representative informatics teachers' education curricula. The ontological models for both curricula have the same top level of competencies model (classes *Knowledge* and *Skills*) and the same relational structure (*hasKnowledge*, *hasSkill*). The task-specific semi-automated method based on standard algorithms for ontology alignment for curricula comparison was proposed, and a software tool prototype was developed supporting the proposed method. Using the software prototype and curricula ontologies, the team of experts consisting of university professors in the field of informatics teacher education, employees of the Education District Offices (Ministry of Education) and teachers teaching secondary school informatics carried out verification of the proposed approach by means of investigation of the compliance of the standardized secondary school curriculum with the teacher education curriculum.

There are two advantages of the proposed curricula model. The first one is machine readable representation of both curricula that facilitates exchange and joint development of curricula, while the second one is its capacity to support representation of the standardized curricula, which is confirmed by ontology representing ACM K12 compliant secondary school curriculum. The constraints are model's capacity to represent some important additional curriculum aspects (instructional design, teaching materials, etc.) and its heavy reliance upon competences not being easy to define unambiguously. The latest is confirmed by experts reporting that the values of similarity, as well as the adequacy of matching, were lower in classes modeling the outcomes/skills of subject areas or courses. Extending ontologies as to comprise other curriculum aspects could alleviate the first constraint, while the second one could be alleviated by better structuring the ontology part that represents skills and/or by utilizing fuzzy ontologies. Future research concerning curriculum model will take these directions.

The main advantages of the proposed curricula harmonization method are the utilization of the standard ontology alignment methods for curricula comparison modified as to exploit the model of competences common to both curricula, and manual intervention option available to experts that could provide for acquiring and integrating deeper experts' knowledge into curriculum model. The need for manual intervention option is already confirmed by independent experts' reports

indicating that some of the class pairs obtained at certain stages do not reflect the real similarity between equivalent concepts in the curricula. The constraints are close coupling of the method with the ontological model and performance issues. The architecture of the matching engine enables simple introduction of other types of matchers (like internal structural similarity or extensional methods) and/or modification of the existing ones in accordance with ontological model thus relaxing the first constraint. One way to improve performance is to apply some procedures for the early elimination of matching candidates. Future research regarding the system's performance will also explore the possibilities of using the approach described in [36]. Last but certainly not least important, a further research direction is the improvement of the evaluation by means of increasing the set of curricula to be evaluated and extending the experts team.

References

[1] Adam, S. (2008) Learning Outcomes, Current Developments in Europe: Update on the Issues and Applications of learning Outcomes Associated with the Bologna Process. *Bologna Seminar: Learning outcomes based higher education: the Scottish experience*. http://www.ehea.info/Uploads/Seminars/Edinburgh_Feb08_Adams.pdf

[2] Fernández-Breis, J. T., Castellanos-Nieves, D., Hernández-Franco, J., Soler-Segovia, C., Robles-Redondo, M. C., González-Martínez, R. & Prendes-Espinosa, M. P. (2012) A Semantic Platform for the Management of the Educative Curriculum, *Expert Systems with Applications*, 39(5), 6011-6019

[3] Dexter, H., & Davies, I. (2009) An Ontology-based Curriculum Knowledgebase for Managing Complexity and Change. *Ninth IEEE International Conference on Advanced Learning Technologies*, *ICALT*, 136-140

[4] Cassel, L., Davies, G., LeBlank, R., Snyder, L., & Topi, H. (2008) Using Computing Ontology as a Foundation for Curriculum Development. *Proc. SWEL@ITS '08: The Sixth International Workshop on Ontologies and Semantic Web for E-Learning*, 21-30

[5] Demartini, G., Enchev, I., Gapany, J., & Cudré-Mauroux, P. (2013) The Bologna Ontology: Fostering Open Curricula and Agile Knowledge Bases for Europe's Higher Education Landscape. *Semantic Web - Interoperability, Usability, Applicability*, 4(1), 53-63

[6] Alatrish, E. S., Tošić, D., & Milenković, N. (2014) Building Ontologies for Different Natural Languages. *Computer Science and Information Systems*, *11*(2), 623-644

[7] Vesin, B., Ivanović, M., Klašnja-Milićević, A. & Budimac, Z. (2013) Ontology-based Architecture with Recommendation Strategy in Java Tutoring System. *Computer Science and Information Systems*, *10*(1), 237-261

[8] Gluga, R., Kay, J. & Lever, T. (2013) Foundations for Modeling University Curricula in Terms of Multiple Learning Goal Sets. *IEEE Trans. on Learning Technologies*, *6*(1), 25-37

[9] Mandić, M., Segedinac, M., Savić, G., & Konjović, Z. (2013) IEEE RCD Standard-based Ontological Modeling of Computer Science Curriculum. *Proceedings of the 3rd International Conference on Information Society and Technology*, 189-285

[10] Cruz, I., Stroe, C., Caimi, F., Fabiani, A., Pesquita, C., Couto, F., & Palmonari, M. (2011) Using AgreementMaker to Align Ontologies for OAEI 2011. *Proceedings of the Sixth International Workshop on Ontology Matching*, 114-121

[11] Jean-Mary, Y. R., Shironoshita, E. P., & Kabuka, M. R. (2009) Ontology matching with semantic verification. *J. Web Semantics*, *7*(3), 235-251

[12] Li, J., Tang, J., Li, Y., & Luo, Q. (2009) RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. *IEEE Transactions on Knowledge and Data Engineering*, *21*(8), 1218-1232

[13] Euzenat, J., & Shvaiko, P. (2007) *Ontology Matching*, Springer-Verlag, Berlin-Heidelberg, p. 333

[14] Shvaiko, P., & Euzenat, J. (2011) Ontology Matching: State of the Art and Future Challenges. *IEEE Trans. Knowl. Data Eng.*, *25*(1), 158-176

[15] Anohina-Naumeca, A., Graudina, V., & Grundspeņķis, J. (2012) Curricula Comparison using Concept Maps and Ontologies. *International Scientific Conference: Applied Information and Communication Technologies*, 5, Jelgava (Latvia), 177-183

[16] Fleishman, E. A., Wetrogan, L. I., Uhlman, C. E., & Marshall-Mies, J. C. (1995) Abilities In Peterson, N. G., Mumford, M. D., Borman, W. C., Jeanneret, P. R. & Fleishman, E. A. (Eds.), *Development of Prototype Occupational Information Network Content Model* , Salt Lake City, UT: Utah Department of Employment Security, Vol. 1, p. 1086

[17] Learning Technology Standards Committee of the IEEE Computer Society (2008) *IEEE Standard for Learning Technology—Data Model for Reusable Competency Definitions*. http://www.doleta.gov/usworkforce/pdf/2007-ieeecomp.pdf

[18] Mirabile, R. (1997) Everything You Wanted to Know about Competency Modeling. *Training and Development*, *51*(8), 73-77

[19] Shippmann, J., Ash, R., Battista, M., Carr, L., Eyde, L., Hesketh, B., Kehoe, J., Pearlman, K., Prien, E., & Sanchez, J. (2000) The Practice of Competency Modeling. *Personnel Psychology*, *53*(3), 703-740

[20] Mandić, M., Konjović, Z., & Ivanović, M. (2015) Ontological Model of the Standardized Secondary School Curriculum in Informatics. *Proceedings of*

the 5th International Conference on Information Society and Technology, 363-367

[21] Krathwohl, D. R. (2002) A Revision of Bloom's Taxonomy: An Overview. *Theory into Practice*, *41*(4), 212-218

[22] Fuller, U., et al. (2007) Developing a Computer Science-Specific Learning Taxonomy. *ACM SIGCSE Bulletin*, *39*(4), 152-170

[23] Gal-Ezer, J., & Stephenson, C. (2010) Computer Science Teacher Preparation is Critical. *ACM Inroads*, *1*(1), 61-66

[24] International Society for Technology in Education (2011) ISTE Standards Computer Science Educators. http://www.iste.org/docs/pdfs/20-14_ISTE_Standards-CSE_PDF.pdf

[25] Association for Computing Machinery (2008) ACM Computer Science Curriculum 2008: An Interim Revision of CS 2001 CS curriculum. http://www.acm.org/education/curricula/ComputerScience2008.pdf

[26] Association for Computing Machinery (2012) The 2012 ACM Computing Classification System. http://www.acm.org/about/class/2012

[27] International Society for Technology in Education (ISTE) (2002) Educational Computing and Technology Standards for Secondary Computer Science Education Initial Endorsement Program. http://www.iste.org/docs/pdfs/ncate_iste_csed_2002.pdf?sfvrsn=2

[28] Churches, A. (2007) Bloom's Digital Taxonomy http://www.techlearning.com/techlearning/archives/2008/04/andrewchurches.pdf

[29] The CSTA Standards Task Force (2011). CSTA K–12 Computer Science Standards. http://csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf

[30] Wu, W., Yu, C., Doan, A., & Meng, W. (2004) An Interactive Clustering-based Approach to Integrating Source Query interfaces on the Deep Web. *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, 95-106

[31] Lin, D. (1998) An Information-Theoretic Definition of Similarity. *Proceedings of the 15th International Conf. on Machine Learning*, 296-304

[32] Jaro, M. (1989) Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida, *Journal of the American Statistical Association*, *84*(406), 414-420

[33] Winkler, W. (1999) *The State of Record Linkage and Current Research Problems*, tech. report 99/04, Statistics of Income Division, Internal Revenue Service Publication

[34] Maedche, A., & Staab, S. (2002) Measuring Similarity between Ontologies. *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, 251-263

[35] Stoilos, G., Stamou, G., & Kollias, S. (2005) A String Metric for Ontology Alignment. In: Gil, Y., Motta, E., Benjamins, V. R., Musen, M. A. (eds.) *ISWC 2005. LNCS,3729*, 623-637

[36] Ehrig M., & Staab, S. (2004) QOM - Quick Ontology Mapping. *Proceedings of the 3rd International Semantic Web Conference (ISWC04)*, 683-697