On Exchanging Ontologies

Christos Tatsiopoulos[#1], Basilis Boutsinas[#2]

[#]Dept. of Business Administration, University of Patras,GR-26500, Rio, Greece

[1]ctatsio@gmail.com

[2]vutsinas@upatras.gr

***Abstract*-Due to the decentralization in the Semantic Web, ontologies can be designed and developed by different communities, using different vocabularies and overlapping content. In this paper, we present a system for ontology exchanging between communities. More specifically, the system updates parts of ontologies, which are considered to be interesting by its designer, using the knowledge included in another. Interestingness is detected automatically utilizing a set of newly proposed measures. Updating is based on a new ontology mapping technique. Also, to facilitate the comparison, we present a survey of ontology mapping and merging/alignment techniques.**

***Keywords*-Information Exchange; Ontology Mapping; Ontology Merging; Ontology Alignment**

I. INTRODUCTION

Ontology engineering which includes designing, developing, maintaining and sharing ontologies, is an emerging knowledge engineering process. It allows the information to be organized into taxonomies of concepts, be represented by attributes, and to disclose relationships between concepts, be represented by functions, IS-A relations, constraints, etc. Ontologies find acceptance in numerous applications, e.g. information retrieval [40], document management [28], agent communication [23], finance [17] and e-commerce [37,49].

Ontologies are imposed by the explosive growth of the Semantic Web, where they are used to describe the semantics of the data. They are used for conceptually structuring of data and for knowledge sharing. Due to the decentralized nature of both the WWW and the Semantic Web [4,50], it is inevitable that different communities, i.e. groups of people with similar interests, within the so-called information society represent and treat the same basic concepts in different ways. For example, the basic concept “Person” is treated entirely different in the medical ontology from that in the business one. Therefore, ontologies can be designed and developed by different communities without adopting common standards for information exchange. On the other hand, the leverage of synergies of information exchange has been increased by the deployment of systems for community interaction support. Many researchers (e.g. [28,29,32,45]) argued that the commonness of all systems ontologies cannot be guaranteed (see [51] for a survey of such effort), because it is more efficient if a smaller community is involved in the process and, in general, communities can usually not be forced to adopt common standards.

In this paper we present a system for ontology exchange between agents, in which ontologies may use different vocabularies and may have overlapping content. This problem is important for agent-oriented applications of ontologies, whenever an agent need to update its knowledge from other agents. The system is based on detecting interesting parts of ontologies and then using a new ontology mapping technique, which is based on association rule mining.

In the rest of the paper, we first describe the technique for detecting interesting parts of ontologies (Section 2) and the new ontology mapping technique (Section 3) and then we present the system architecture (Section 4). Finally, we draw a conclusion (Section 5).

II. ON DETECTING INTERESTING PARTS OF ONTOLOGIES

Within the proposed system, we consider the problem of exchanging parts of ontologies but not the whole ontologies, which is actually related to ontology merging and alignment. More specifically, we consider the problem of updating an ontology using the knowledge included in another. However, the update is performed only for the parts of the first ontology which are considered interesting by its designer. The interestingness is decided by the system itself but not the designer.

Thus, we reduce the problem to automatic detection of the interesting parts based only on the structure of the ontology and not on any user input. Of course, the knowledge included in the interesting parts of the one ontology must be a superset of knowledge included in the interesting parts of the ontology to be updated.

To tackle the problem, we define a number of different measures of interestingness of parts of ontologies. Each such measure represents different semantics of interestingness. After the interesting parts have been located in any alignment, the algorithm proposed in the literature could be applied (see next subsection) for updating the ontology.

The term "interestingness" was first used in data mining as a measure of how much interesting an extracted data mining rule is with respect to a user judgement. "Entropy" and "support" are such measures of interestingness. There is not any generally accepted definition of interestingness. In fact, each of the proposed measures concerns a different aspect of what "interestingness" in ontologies could mean. Note that the proposed measures exploit only the structure of the ontologies, the proposed measures are domain/application independent.

A. Related Merging/Alignment Techniques

There are some techniques presented in the literature for ontology merging and alignment. They all consider the merging or alignment of the entire input ontologies. The key ideas of any of them could be applied to the alignment of interesting parts after they have been located

The methodology presented in [20] merges mapped concepts and then it applies validation heuristics in order to avoid both structural and content-based inconsistencies. For instance, it checks: 1) for new superconcepts of a merged concept, i.e. concepts that were not already superconcepts of the merged concept in the input ontology, 2) if mutual disjointness requirements are still satisfied by merged concepts, as in the input ontologies, 3) if any cycles are created after merging, 4) if there are any type constraints violations after merging.

Chimaera [31] is a tool that focuses the attention of the user in particular portions of the ontology that are semantically interconnected and in need of repairing or further merging. More specifically, Chimaera coalesces semantically identical terms from different ontologies, so that they are referred to by the same name in the resulting ontology, and identifies terms that should be related by subsumption, disjointness, or instance relationships and provide support for introducing those relationships. After merging, Chimaera provides support for recognizing logical inconsistencies and for validating the resulting ontology.

FCA-Merge [45] employs Formal Concept Analysis to create, in a first phase, mappings between populated ontologies. It is based on a set of natural language documents. It represents the set of documents assigned to each inputted ontology using the vector space model, it generates two formal concepts out of these sets and it generates a pruned concept lattice. The merging phase is performed after processing the formal concepts of the pruned concept lattice. Each such concept is a candidate for a concept, a relation, or a new subsumption in the merged ontology. The user has to interact with the system in order to resolve possible conflicts and duplicates, but there is automatic support from FCA-Merge in terms of a query/answering mechanism based on heuristics. Several cases are examined. For example, two or more concepts of the source ontologies generate the same formal concept. This indicates that the concepts should be merged into one concept in the target ontology. The user is asked which of the names to retain.

The PROMPT tool [34] is designed to maintain the focus of users and to provide feedback to the user during mapping and then during merging or alignment. First PROMPT creates an initial list of linguistic similarity matches based on class names. Then, the user triggers an operation which either has been suggested by PROMPT or by using an ontology-editing environment. Then, PROMPT performs the operation and it automatically executes additional changes, generates a list of suggestions, and determines conflicts. There are specific such operations for merging/alignment. For example, the operation "perform a deep copy" includes copying all the parents of a class down to the root of the hierarchy and all the classes and slots it refers to. After such operations, conflicts that may appear could be: 1) name conflicts (more than one frame with the same name), 2) dangling references (a frame refers to another frame that does not exist), 3) redundancy in the class hierarchy (more than one path from a class to a parent other than root), etc.

Ontomorph [9] is a tool to support the translation of symbolically represented knowledge into some different form. It is based on a mixture of syntactic and semantic criteria. It can be used to support knowledge-based merging tasks, as well. This is accomplished by finding, at first, semantic overlaps and then by designing transformations to bring sources into mutual agreement. Finally, it checks the resulting knowledge for consistency, uniformity, and non-redundancy. After checking, the previous steps will be repeated if it is necessary.

OntoDNA [27] is an automated ontology mapping and merging tool. It utilizes Formal Concept Analysis to capture the properties and the inherent structural relationships among ontological concepts of heterogeneous ontologies. More specifically, it generates two formal concepts out of one source and one target ontology. After a pre-linguistic processing of these formal concepts, certain mapping rules are applied to reconcile their intents. Thus, the captured structures of ontological concepts act as background knowledge to resolve semantic interpretations in the next phase, where unsupervised clustering techniques, (Self-Organizing Map and K-means), are used. Finally, the ontological concepts of the target ontology are updated to the source ontology based on certain merging rules.

In [44] the alignment problem is reduced to the discovery of subsumption relations among ontology elements. The proposed technique computes subsumption relations between concept pairs of the two input ontologies extracted by a mapping tool. Thus, concept pairs are represented as feature vectors of length equal to the number of the distinct properties of source and target ontologies: properties with equivalent meaning correspond to the same vector component. Then, given a pair of concepts, a supervised machine learning method locates a hypothesis concerning their relation in a space of hypotheses, which best fits to the training examples, generalizing beyond them. The training examples for the learning method are being generated from the target and source ontologies.

In [5] a semi-automatic merging algorithm is presented, where users can choose appropriate results from a set of suggestions. It is based on combining Answer Set Programming with linguistic background knowledge. The latter is used to detect correspondences between concepts based on synonymy while Answer Set Programming calculates several answer sets which provide merging options among which the user can choose. To restrict the exponential number of possible merging solutions to reasonable ones, there are constraints in the form of intuitive merging rules which the user can apply.

Finally, a similar problem is that of database schema integration (e.g. [43]). However, most schema matching and integration techniques are not adequate for ontology mapping and merging/alignment due to the not handled differences in terminology, exhibiting poor results in the case of little structural similarity, and absence of instances, etc. Despite the support or the controversy of the statement that ontology mapping/alignment is similar to database schema matching/integration [25,36], the proposed measures could be applied to both of them.

B. The Proposed Interestingness Measures

We propose some measures of interestingness and we present their definitions. We have investigated several others. However the proposed ones exhibit significant results during empirical tests. They are all based on the structure of ontologies.

The proposed measures are all based on detecting interesting concepts within ontologies. Then, we consider interesting parts of an ontology the subgraphs rooted in these interesting concepts. Thus, the proposed measures assign a value to each concept representing its interestingness:

rel_cD% Percentage Direct Child Nodes: the number of nodes which are directly connected to a specific node, as percentage of the number of nodes in the ontology. Note that the higher is the value of the measure for a node, the greater is its interestingness. For example, in ontology shown in Fig. 1, node oo250 (rel_cD%=10.71%) is more interesting than oo180 (rel_cD%=0%) or even ooo40 (rel_cD%=7.14%), since it has more direct child nodes.

rel_cI% Percentage Indirect Child Nodes: the number of nodes in the subgraph rooted at a specific node, as percentage of the number of nodes in the ontology. Note that the higher is the value of the measure for a node, the greater is its interestingness. For example, node ooo95 (rel_cI%=42.86%) is more interesting than oo185 (rel_cI%=14.29%), since there are more nodes in the subgraph rooted at ooo95.

rel_b% Percentage Brother Nodes: the number of Direct Child Nodes of the father node(s), i.e., of immediate ancestor(s), of a specific node, as percentage of the number of nodes in the ontology. Note that the higher the value of the measure for a node, the greater its interestingness. For example, node oo250 (rel_b%=7.14%) is more interesting than node oo180 (rel_b%=3.57%) and node ooo90 (rel_b%=3.57%), since it has more brother nodes having both oo175 and oo170 as father nodes.

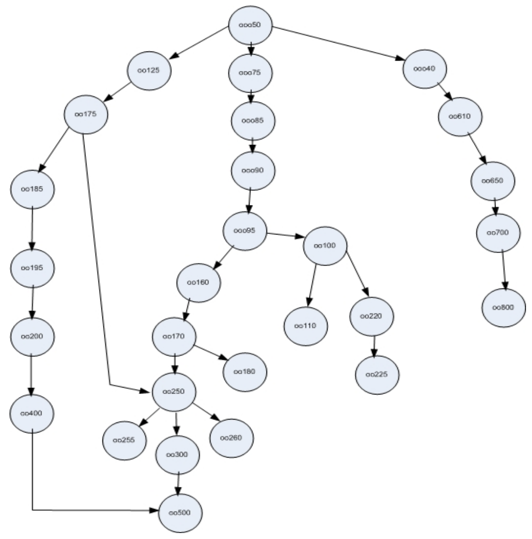

Fig. 1 A test ontology

mdisbr Mean Distance of Brother Nodes: the mean distance of a specific node from its Brother Nodes. The distance of two nodes d(x,y) is calculated using the dissimilarity measure presented in [6]. The dissimilarity between any two attribute values is represented by the distance between the corresponding nodes of the tree structure as defined by the following formula:

$d(X,Y)=d_1*d_2*d_3$, where

$d_1=1/fl(X,Y)$,

$d_2=Average((l(X)-fl(X,Y))/max(p(X)),$

$(l(Y)-fl(X,Y))/max(p(Y)))$ and

$d_3=p(X,Y)/(max(p(X))+max(p(Y)))$,

Where X and Y represent any two nodes, fl(X,Y) is the level of the nearest common father node of X and Y nodes, i.e. the level of the nearest common predecessor, l(X) is the level of node X, i.e. the depth of the node, max(p(X)) is the length of the maximum path starting from the root to a leaf and containing node X, p(X,Y) is the length of the directed path (number of edges) connecting X and Y. If there is not a path connecting X and Y then p(X,Y)=p(X,fl(X,Y))+p(Y,fl(X,Y)). Also, p(X,X)=0. mdisbr is calculated by the following algorithm:

```
for each Brother Node i of node j
 calculate d(X,Y)
 set count += 1
 set dsum += d(X,Y)
return dsum/count
```

Note that the lower is the value of the measure the greater is the interestingness. For example, node oo110 (mdisbr=0.003) is more interesting than node oo610 (mdisbr=0.0115), since its father node is located deeper in the ontology.

nden(k) Network Density of range k: Network Density of range k of a specific node i is the number of nodes that are connected to or can be reached from i, via a path of length at most k, which does not include direction changes. We have implemented the calculation of the measure dynamically. Note that the higher is the value of the measure for a node, the greater is its interestingness. For example, for node oo610 nden(2)=4, since there are 2 ancestor nodes (ooo40,ooo50) and two successor ones (oo650,oo700). Thus, it is more interesting than node oo110 (nden(2)=2).

in% Percentage Incoming Paths: the indegree (din(i)) of a node i, i.e., the number of edges which have i as their end-node, as percentage of the total incoming and outcoming paths, i.e., of the indegree plus outdegree of i. Note that the higher is the value of the measure for a node, the greater is its interestingness. For example, node ooo90 (in%=66.67%) is more interesting than node oo250 (in%=40%).

out% Percentage Outcoming Paths: the outdegree (dout(i)) of a node i, i.e., the number of edges which have i as their start-node, as percentage of the total incoming and outcoming paths, i.e., of the indegree plus outdegree of i. Note that the higher is the value of the measure for a node, the greater is its interestingness. For example, node oo250 (out%=60%) is more interesting than node ooo90 (out%=33.33%).

n_l(i)% Percentage Level distribution: Level distribution of a specific node i is the number of nodes belonging to the level of node i (l(i)), (i.e., the length of the maximum path (number of edges) from the root to node i), as percentage of the number of nodes in the ontology. Note that the higher is the value of the measure for a node, the greater is its interestingness. For example, there are 4 nodes on the same level with node oo100 (out%=14.3%) which is more interesting than node oo250 (out%=10.7%), since there are 3 nodes on its level.

We evaluated the proposed interestingness measures on both test and real ontologies (e.g. the Gene Ontology (http://www.geneontology.org/). To evaluate the proposed measures we used a node ranking with respect to their interestingness, defined by human experts, i.e. researchers of the "Institute for the Language Processing – ILSP" in Greece (www.ilsp.gr). After testing several test ontologies, we can conclude that rel_cD%, nden(k), out%, and n_l(i)% measures assigned interestingness in a way that reflects expert's first choices [47]. Moreover, both the n_l(i)% and the out% measures discovered additional interesting nodes. Such results were returned to the experts for additional comments and most of them were accepted as valid according to expert's criteria. However, the rest measures have not provided successful results in a consistent manner.

We have also investigated the integration of the different measures by introducing a unified model M = w_1*rel _cD%, w_2*rel_cI%,w_3*rel_b%,w_4*mdisbr, w_5*nden(k), w_6*in%, w_7* out%, w_8*n_l(i)%, as a function of them, where w_i is a weight. Experimental tests show that a different unified model is needed for each different type of structure. In Table I, weights for four different types are shown resulting in about 95% accuracy w.r.t. expert, on the average.

TABLE I
WEIGHTS FOR DIFFERENT TYPES OF ONTOLOGIES

Type	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8
balanced tree	8.5	6.8	10.2	13.0	13.0	33.6	5.5	9.4
unbalance d tree	3.3	7.7	11.0	13.6	5.8	49.5	5.7	3.3
sallow tree	18.2	17.5	1.1	13.9	14.5	27.8	5.9	1.1
balanced graph	10.6	0.7	12.7	13.0	4.5	6.7	9.4	42.3

The notion of interesting parts in ontologies has already mentioned in the literature. It is mainly applied to web pages (e.g. [11]). It is also related to the problem of selecting ontologies under certain constraints [42] or to search engines [8]. Finally, the notion of interestingness is also applied to database schemata [3].

The most related to the proposed notion of interestingness is the notion of \importance of concepts and relations" which is based on a measure similar to out%, although it is a more sophisticated one [52]. Also, very closely related is the notion of "degree-of-interest" [15], where concepts of interest are detected by users based on their navigation activities.

III. A NEW ONTOLOGY MAPPING THECHNIQUE BASED ON ASSOCIATION RULE MINING

Ontology mapping aims at tackling structural and semantic heterogeneity and incompatibility by determining correspondences among elements of disparate ontologies. Note that structural heterogeneity has also been addressed to a great extent in the schema matching literature [41]. A mapping can be established either directly between two ontologies (alignment) or indirectly through mapping them onto a third reference ontology which both of them share as a common upper model (articulation).

The work of mapping ontologies is performed mostly by hand, perhaps supported by a graphical user interface. Of course, performing ontology mapping manually is an extremely time-consuming and error-prone process. The ontology mapping techniques presented in the literature are usually based on syntactic and/or semantic heuristics. The latter have been studied in various scientific fields including machine learning, concept lattices, formal theories, databases and linguistics. In almost all of them user intervention is required, thus they are semi-automated. Usually, when an automatic decision is not possible, these techniques suggest possible correspondences, determine conflicts and propose solutions and actions. Then the user makes the final selection.

We present a new ontology mapping technique (the ONARM technique) which, given two input ontologies, is able to map concepts in one ontology onto those in the other without any user intervention. The proposed technique exploits the structure of the input ontologies, i.e. the concept hierarchies, to determine the mapping. More specifically, it is based on association rule mining in order to extract association rules from these concept hierarchies. Association is one of the most popular data mining tasks. Association rules can be used to represent frequent patterns in data in the form of dependencies among concepts-attributes. The extracted association rules are considered as indirectly describing the concept relationships.

A. Previous Approaches

Recently, the number of ontology mapping techniques and systems has increased significantly (see http:// www. ontologymatching.org for the complete information on the topic).

Ontology mapping techniques vary in input and output formats as well as in modes of user intervention. There has been little work on the comparative evaluation of ontology mapping techniques in the literature (e.g. [19,25,26]). Next, we present related techniques focusing on the mode of user intervention.

FCA-Merge [45] can semi-automatically create mappings among populated ontologies using a set of natural language documents, which have to be relevant to each of the input ontologies. Concepts which cannot be matched have to be treated manually (or the set of documents has to be expanded). During the last phase, the user manually, with guidance from FCA-Merge, constructs the merged ontology. This construction is semi-automatic as it requires background knowledge about the domain. The engineer has to resolve possible conflicts and duplicates, but there is automatic support from FCA-Merge in terms of a query/answering mechanism based on heuristics, which aims at guiding and focusing on the engineers attention on specific parts of the construction process.

CAIMAN technique [28] employs information retrieval methods to apply to corpus of relevant documents that are assigned to each inputted ontology, so as to automatically create the mappings. For each concept in the first ontology, CAIMAN calculates a probability measure for any concept of the second ontology to be the corresponding node. The calculation of the probability measure is based on a simple cosine measure of the feature vectors of the two concepts.

GLUE technique [12] employs machine learning methods to semi-automatically create semantic mappings between populated ontologies. At first, for each concept within ontology, GLUE finds the most similar concept in the other ontology. Similarity is defined as the joint probability distribution (e.g. Jaccard coefficient) of the concepts with respect to the instances of the ontologies. In order to calculate the similarity, Glue uses ensemble classifiers. Finally, GLUE attempts to exploit available domain constraints and general heuristics in order to improve mapping accuracy. User intervention is required when concepts cannot be matched (3-34% of the concepts on several real-world domains).

IF-Map technique [24] employs information flow theory to automatically create mappings between populated ontologies. IF-Map finds logic infomorphisms, if any, between the input ontologies and displays them in RDF format. Logic infomorphisms represent the information flow between separate components of a distributed system. Components are represented by local logics which describe the different vocabularies, i.e. the dfferent instances (along with their different constrains) and their classifications to different types, used in each component.

ONION technique [32] employs linguistic and structure-based heuristics to determine the matches, i.e. the “articulation rules”, and finally to semi-automatically create the mappings. Mappings in articulation are established indirectly through mapping concepts of input ontologies with a shared view, which is created on purpose. Linguistic matches are based either on dictionaries like WordNet or on a corpus of documents related to the input ontologies. Structure-based matches are searched after the linguistic matches. Finally, the suggested matches are validated by the user. Also, ONION includes a learning component in the system which takes advantage of users’ feedback to generate better matches in the future for similar ontologies.

The mapping technique of the ITTalks system [39] employs either a statistical majority- based heuristic or a text classification algorithm in order to test all possible matches or to test user defined matches. According to the heuristic, if a concept in the first ontology matches with the majority of the children of a concept in the second ontology, then the latter concept is a better mapping than its children. The text classification algorithm is used to build a classifier for each concept in the ontologies. The classifier is based on a set of exemplar documents assigned exclusively to each concept.

EER-CONCEPTOOL technique [10] employs a Description Logic in order to build an articulation view for mapping the input ontologies. It performs a number of steps of formal concept and linguistic analysis in order to suggest the user possible correspondences. However, in each step the user is ultimately responsible for accepting or rejecting each of these suggestions for concept, attribute or identifier correspondences. Moreover, the user can introduce more complex correspondences between unions or intersections of concepts in each ontology, based on the ones proposed by the system.

MAFRA technique [30] employs semantic similarity measure and heuristics based on the similarity of two concepts to establish a matching. It uses a semantic similarity measure which is based on hypothesis that the more the information two concepts share in common, the more similar they are. The information shared by two concepts is indicated by the information content of their most specific common subsumer. After similarities have been established the “semantic bridging” heuristic phase is responsible for i) establishing correspondence between concepts, attributes and relations, ii) endowing the mapping with bridges for concepts that do not have a specific counterpart target concept.

S-MATCH technique [18] concentrates on semantic matching and it employs linguistic and logic based heuristics to automatically create the strongest semantic relation between every pair of concepts in the input ontologies. It distinguishes between a concept related to a label (what the label means to the world) and a concept related to a node (what means the concepts of labels assigned to the nodes above it). Thus, in the preprocessing phase all labels are translated into an internal language, which is based on propositional logic after lexical processing. Then, it computes relations between every pair of concepts of labels (CL matrix) based on two sources of a priori knowledge: the one extracted from word similarities by matchers using string manipulations and that extracted from WordNet by matchers using the “senses”. Then, it computes relations between every pair of concepts of nodes (CN matrix) using CL by a satisfiability problem solver for propositional logic.

OLA technique [16] employs semantic similarity between concepts, which is based on the similarities of concepts (contributor pairs) related to those are going to be checked (anchor pair). It uses different semantic similarity measures to calculate the similarity of a pair. To one of the eight supported elements of an ontology, i.e. categories (e.g. class, object, relation, etc.), a different similarity measure is defined. All of them form the final similarity by using weights.

There are also some techniques that aim at guiding the user to create mappings. PROMPT [34], (as well as SMART [33] and PROMPTDIFF [35]), employs linguistic similarity and heuristics to guide the user in the creation of the mapping. Firstly PROMPT creates an initial list of linguistic similarity matches based on class names. Then, the user triggers an operation which either has been suggested by PROMPT or by using an ontology-editing environment. Then, PROMPT performs the operation and it automatically executes additional changes, generates a list of suggestions, and determines conflicts. CHIMAERA [31] is a similar interactive browser-based editing, merging, and diagnosis tool for the Ontolingua editor. As in PROMPT, the user is in charge of making decisions that will affect the mapping process. CHIMAERA analyzes the ontologies which are to be merged, and if linguistic matches are found, the merge is done automatically; otherwise the user is prompted for further action.

Moreover, there are techniques which are based on the combination of different mapping processes (e.g. [2,21]), which exhibit remarkable results in terms of accuracy (http://oaei.ontologymatching.org). Also, there are techniques that could potentially be used in ontology mapping like translators (e.g. OntoMorph [9]) or integrators (e.g. 20). Finally, a similar problem is that of schema matching in databases.

B. The Proposed Ontology Mapping Technique

The key idea of the proposed ontology mapping technique is to establish a similarity between two concepts of the input ontologies, which is based on their location in the ontology structures. The location of a node, which represents a concept within an ontology structure, determines its neighbour concepts. We consider that the meaning of a concept is also characterized by the meaning of its neighbour concepts, as the creator of the ontology indirectly determined by defining the structure of the ontology.

Note that structural mapping alone is not sufficient for ontology mapping. The meaning of the concept is also characterized by a linguistic analysis of the concept with respect to a large-scale dictionary like WordNet, to a corpus of documents, to manual rules, to lexical distances, etc. The proposed technique accepts both of these sources of background knowledge in order to establish a similarity measure. However, the latter is dominated by the location of a concept within the ontology.

Graph matching techniques could be used in order to examine the similarity of the location of two input concepts. Since we concentrate on efficiency, we rejected such techniques because of their time complexity (for instance time complexity of graph isomorphism is exponential). The proposed technique considers each path of the ontology structure as a source of background knowledge. It applies association rule mining in order to determine the predominant neighbour concepts of an input concept.

In this paper, we consider association rule mining which is known as the market basket problem, and in which concepts-attributes represent products and the initial database, is a set of customer purchases (transactions). This particular problem is well-studied in data mining. We consider association rules an analog to the form "90% of the customers that purchase product x also purchase product y" (Boolean association rules) (e.g [1,7,38]). Formally, an association rule is a rule of the form $X \rightarrow Y$, where X,Y named respectively antecedent and consequent of the rule and $X,Y \subseteq I=\{i_1,i_2, ... ,i_j\}$, such that $X \cap Y=\emptyset$ and i_k, $1 \leq k \leq j$ is an item in the transaction database D. The informative power (named interestingness) of each association rule is measured by two indexes: the Support that measures the proportion of transactions in D containing both X and Y and the Confidence that measures the conditional probability of the consequent given the antecedent.

More specifically, the proposed technique considers each path of the ontology structure as a transaction. Then, for each inputted ontology, it applies association rule mining to the set of its transactions. We consider that the extracted association rules determine the predominant neighbor concepts of every input concept. Thus, the similarity of these association rules defines the location-based similarity of the concepts.

Linguistic analysis is also taken into consideration. However, it is used to increase or decrease the obtained location-based similarity (see γ parameter below). In that sense, any heuristic for linguistic analysis proposed in the literature can be used. Also, aggregating the results of such heuristics could also be used, as for example in [13,14]. In this paper, we adopt a naive such heuristic: we examine identity of labels of concepts, while we use a common vocabulary for both ontologies. Obviously, more advanced heuristics would increase the overall accuracy.

The proposed technique can be applied to ontology structures forming a directed acyclic graph. Thus, it supports multiple inheritance. The required formal definition of input ontologies contains two core items shared by most formal definitions of an ontology in the literature: concepts and a hierarchical IS-A relation. Thus, we define a core ontology as: a pair G=(C,r), where C is a set of concepts and r is a partial order on C, i.e. a binary relation $r \in CXC$ which is reflexive, transitive, and antisymmetric.

More specifically, the proposed technique accepts two ontologies as input. Any ontology editor can be used to create them (we used the Protégé knowledge-modeling environment). During the first step the input ontologies are transformed to RDF and RDFS formats. Obviously, any ontologies pre-described in RDF(s) can be used. Then, the Java-Jena API is used. Jena is a Java implementation of an extension to the semantic web by means of a respective API. This offers the capability of getting the complete description of the input ontology in terms of its structural elements (paths, current nodes, successor nodes, parent nodes, siblings and leaf nodes). In order to apply the APRIORI association rule mining algorithm [1], nodes must be topologically numbered. Thus, the second step is to generate a numbered node structure, of the same structure as the ontology under examination but numbered with integer numbers that will undergo Breadth First Search (BFS), such that, integers are horizontally incremented and assigned and therefore guaranteeing this way, that any node N_i is numbered with an integer k such that k>m, where m is the integer which has been assigned to its parent node M (this relation holds true for any parent and child nodes within the input ontology). In this way, nodes at deeper levels are mapped with higher number values.

After numbering the Ontology, a hash table is built that includes all the nodes of the ontology with their respective integers. Then, to extract all the possible paths, with the objective to quick reach and examine priority terminal nodes its paths, a Depth First Search (DFS) is run, that provides all possible paths number-named in a list type format. The methodology has been designed in such a way that permits multiple inheritance (and therefore has multiple parents) in the following way and under the definition: L(i) is the level of node i and $L(i)=\max(L(l_1),L(l_2), ... ,L(l_n))$, where $l_1,l_2, ... ,l_n$ are parent nodes of node i.

To resolve the above problem during the numbering process, the integers that are assigned to the nodes are non-continuous but they retain the necessary property needed for the APRIORI algorithm, such as: for any two nodes I and J, order (J)>order (I), if node J is a parent node of I.

This step involves the extraction of all root-to-leaf paths available in the ontology schema by means of a recursive method. Furthermore, a list of all leaf nodes is created. Then, for a predefined set of minimum support and confidence values, APRIORI algorithm is applied to both input ontologies (e.g. G1 and G2). The result is a set of rules of the following type:

1: [2,7], 45, 20
2: [1, 7, 4], 30, 70
3: [1, 3, 6], 45, 20
4: [1, 3, 4], 30, 70
…

where, the integers above denote number-named nodes of the ontology. Each pair of (c,s) produces such one respective set of rules R(G1), R(G2). Following, the above rule set is back translated and represented with the original node names, providing this way R'(G1), R'(G2) of rules.

The proposed technique generates an [nxm] "significance matrix" containing the significance in mapping every node of $G_1=(C_1,r_1)$ with every one of $G_2=(C_2,r_2)$. Note that G_2 is mapped against G_1 and not vice versa, considering G_1 as our reference ontology. The significance in mapping $X \in C_1$ to $Y \in C_2$ is calculated based on the support measure of the association rules having X and Y as left part. For instance, consider the following rules for X and Y:

$X \rightarrow (B,s_1,c_1)$,

$X \rightarrow (BC,s_2,c_2)$,

$Y \rightarrow (B,s_3,c_3)$,

$Y \rightarrow (AC,s_4,c_4)$

For each of the four pairs of rules, one for X and the other for Y , the measures K and K_t, indicating the significance, are computed by the following procedure:

$K=0$, if $|s_X-s_Y|>\alpha$,

where α a user defined threshold.

$K \leftarrow Average(s_X,s_Y)*\beta*w$,

where $\beta>1$ if X or Y or both are instances,

$\beta=1$ otherwise,

w=percentage of similarity of right parts.

$Kt=K*\gamma$,

where $\gamma>1$ if $X\equiv Y$, i.e. the two nodes are identical after a linguistic analysis,

$\gamma=1$ otherwise.

Thus, for the pair:

$X\rightarrow(B,s_1,c_1),Y\rightarrow(B,s_3,c_3)\Rightarrow K=Average(s1,s3)*1*1$, while for the pair:

$X\rightarrow(B,s_1,c_1),Y\rightarrow(AC,s_4,c_4)\Rightarrow K=Average(s_1,s_4)*1*0$ and for the pair:

$X\rightarrow(BC,s_2,c_2),Y\rightarrow(AC,s_4,c_4)\Rightarrow K=Average(s2,s4)*1*0.5$.

Considering that nodes of reference ontology G_1 are listed in rows and those of G_2 are listed in columns, the "significance matrix" is filled as follows:

```
for every cell (i,j) in the n x m matrix
calculate $\Sigma_p K$, $\forall$ pair p of rules, one for i and the other for j
calculate $K_t$
fill cell (i,j) with $K_t$
repeat
find the cell (i,j) filled with the first maximal $K_t$ in the matrix
map i to j
delete row i and column j
until there is not any row left
```

For some minimum support and confidence (e.g. (s,c)=(25,5)), the proposed technique extracts rules from two ontologies (e.g. G_1 and G_2) and builds the significance matrix. Finally, it provides final mappings along with their significance, e.g.:

$[G_1]$: ooo50 $\Rightarrow$ [G2] : ooo50; 75.0

We performed extensive empirical tests (see [46]) aiming at examining the accuracy of the proposed technique using both test and real ontologies. High accuracy is achieved in most of the cases (up to 100%). High accuracy is also achieved on specific OAEI-2008benchmarks (http://oaei.ontologymatching.org/). For instance, 100% accuracy is achieved by testing the ontology 101 of OAEI-2008 against itself since all the 93 nodes were mapped correctly. Also, 100% accuracy is achieved by testing the ontology 101 against ontology 228, since all the 93 common nodes were mapped correctly.

The empirical tests show that minimum support and confidence values are not critical. The only requirement is to set low values. Theoretically this is true because of the small number of paths of an ontology. For instances, the results on the OAEI- 2008 benchmarks mentioned in the previous paragraph were performed setting both minimum support and confidence to 5.

C. Efficiency in Ontology Mapping

The time complexities presented in the following were considered for the worst case.

In the proposed technique, the extraction of association rules is performed separately for each ontology, which is $O(|G_i|*|C_k|)$, where $|C_k|$ is the number of all candidate itemsets checked. According to [1], the complexity of locating the itemsets of size k is $O(k*log(|G_i|/k))$ (however in random databases there are only a few large itemsets). In the proposed technique the maximum size is d, the depth of the ontology. Thus, latter the complexity is: $O(\sum^d_{j=1} j*log(|G_i|/j))$ which is $O(d^2log(|G_i|))$. Thus, the overall complexity of extraction is $O(d^2log(|G_i|))$. The used similarity matrix is obtained in $O(|G_1|*|G_2|)$.

In FCA-Merge, the problem of computing concept lattices has exponential worst case complexity. In practical cases, the "Next Closure" algorithm for computing concept lattices is $O(B(K)*|D|*(|G_1|+|G_2|)^2)$, while the complexity is $O(B(K)*(|D|+(|G_1|+|G_2|))*|D|)$ for the algorithm of Nourine and Raynaud. Also, note that the time complexity of TITANIC algorithm is between them.

CAIMAN technique requires a time-consuming phase in order to build the feature vector of each concept in the input ontologies. The calculation of a similarity matrix for any two concepts, based on the feature vectors, is $O(|G_1|*|G_2|)$.

GLUE is a time consuming technique. For every pair of concepts (A,B) $A\in G_1$, $B\in G_2$ it computes their similarity by applying the Naive Bayesian classifier (assuming that all the base learners an trained in parallel), which is $O(|G_1|*|G_2|*(|n_1|+|n_2|))$, where n_1, n_2 are the instances of G_1, G2 respectively. This is because the complexity of Naive Bayesian classifier is linear to the training set. Also, the complexity of "relaxation labeling", which depends on the compatibility coefficients, is linear to the number of labels, thus it is $O(|G_1|+|G_2|)$.Note also, that the convergence properties of "relaxation labeling"are not yet well understood and it is liable to converge into a local maxima.

IF-Map has an exponential time complexity and thus it cannot be applied to large-scale ontologies. This is because it must automatically generate all possible mappings among ontologies. Thus, the search space is exponential and IF-Map tries to reduce it by using the instances. By using the constraint, a mapping has to respect concept hierarchy.

In ONION technique, linguistic matches which are based on dictionaries require an exhaustive search in extremely large-scale dictionaries with many hundreds and thousands of concepts, like WordNet with 10^6 concepts. Thus, a word similarity construction based on WordNet is $O(|G_1|*|G_2|*|G_{WordNet}|)$. Linguistic matches based on a corpus of documents require a time-consuming preprocessing in order to build the context vector of each concept in the input ontologies. The calculation of a similarity matrix for any two concepts based on the context vectors, is $O(|G_1|*|G_2|)$. Also, the graph isomorphism is a NP problem.

The mapping technique of the ITTalks system requires every pair of concepts either to test their children or to classify the assigned concept exemplar documents. Thus, its complexity is $O(|G_1|*|G_2|*D*O(classify))$, where D is the mean number of exemplar documents assigned to a concept and O(classify) and is the complexity of the classification of a document, a rather complex process.

In EER-CONCEPTOOL technique different inference mechanisms on the Description Logic are used in order that the formal concept and linguistic analysis can be performed.

Although ontologies are represented using the Description Logic, without loss of generality, a formal concept analysis is reduced to traversals of ontology structures, for every concept of the articulation ontology G_a, which is $O(|G_1|+|G_2|*|G_a|)$. Moreover, linguistic analysis requires every concept in the input ontologies, a search in the WordNet, an extremely large-scale dictionary with 10^6 concepts, which is $O(|G_1|+|G_2|*|G_{WordNet}|)$.

In MAFRA technique the time complexity of the adopted semantic similarity measure for two concepts is O(d), where d is the depth of the ontology. Thus, the time complexity of calculating a similarity matrix is $O(|G_1|*|G_2|*d)$. The semantic bridging phase is reduced to access the similarity matrix and to perform ontology traversals, which is not significantly more time-consuming than the calculation of the similarity matrix.

In S-MATCH technique the most time consuming phase is the computation of the C_N matrix. For every pair of concepts of nodes a satisfiability problem solver must be applied. The satisfiability problem is a known NP-complete problem requiring an exponential time. The overall complexity of this phase is $O(|G_1|*|G_2|*2^V)$, where V is the number of logical variables in the propositions represented the background knowledge.

In OLA technique, for each pair of concepts an iterative process is executed in order to calculate similarity. Complexity is increased if there are recursive dependencies (two pairs of nodes are each other's contributor). In general, a non- linear problem must be solved in order to calculate similarity.

Finally, note that recently there is a concern on efficiency of the proposed ontology mapping techniques in the literature. For instance the work presented in [13] tries to reduce the search space by introducing certain strategies to select the pair of concepts checking for mapping. Thus the time complexity is reduced to $O((|G_1|+|G_2|)*\log(|G_1|+|G_2|))$.

D. System Architecture

A system is implemented, the Concept Net, in order to evaluate the proposed methodology in practical applications. The general system architecture is composed by two major modules. The first module, the front-end module, runs on user's personal computing device. Note that the system can run on mobile devices such as Personal Digital Assistants and mobile phones. The second module, the back-end module, implements all the logic and algorithms and it runs on the server side. These two modules communicate to each other via web services which have been developed for this purpose.

The front-end module (Fig. 2) is composed by three main components: the "Graphical User Interface", the "Web Services Interfaces (WS IFCs)" and the "Operating System Utility".

The "Graphical User Interface" component consists of a set of forms via with the end user interacts with the system. It includes all the forms that the user uses to interact with the system, as well as the forms that compose the GUI for the developed ontology editor. The ontology editor is a light graph editor that allows the end user to visualize all the concepts that are relevant to him, as well as their relations, in a form of a graph. This graph is stored locally to this module.

The "Web Services Interfaces (WS IFCs)" component consists of service calls, which are responsible for all the communication between the client front-end device and the back-end system. Such kind of data is low level data, like the physical location where the main server is located in terms of IP, port, function calls, etc. This component can send and receive specific user information, like authentication information, graph to be processed, reception of similar nodes from other nodes from other users in the Concept Net, etc.

The "Operating System Utility" component consists of low level utilities that organize functionality dealing with access to file system. This functionality incorporates storage and retrieval of users' credentials, personal infomation regarding to the system accessibility, storage of graphs/ontologies that the user creates and graphs which are received from the back-end system. All the communication between the client system in the front-end device and the server based back-end system is achieved via this set of web services calls only.

The back-end module (Fig. 3) is currently implemented using the Java Programming Language. It is composed by the "Web Services Interfaces (IFC)", the "Data Access Layer", the "System Logic & Intelligence Module", the "Knowledge Service", the "Database" and the "Operating System & File System Access Layer".

The "Web Services Interfaces (IFC)" component is responsible for all the communication between the users' client devices and the back-end system.

The "Data Access Layer" incorporates all the functionality needed to send queries to system database as well as to receive information from it, during the system operation. This information is relevant to authentication as well as to user's knowledge profile.

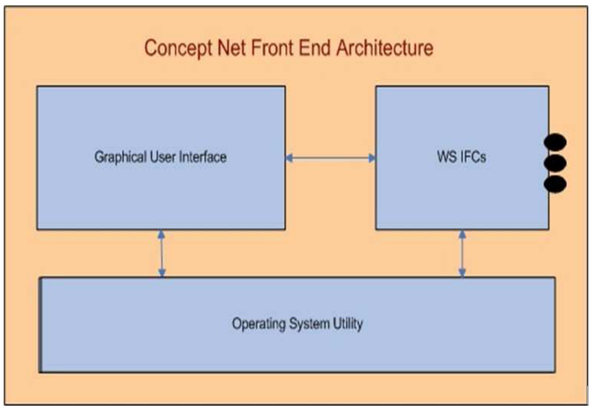

Fig. 2 The front-end module

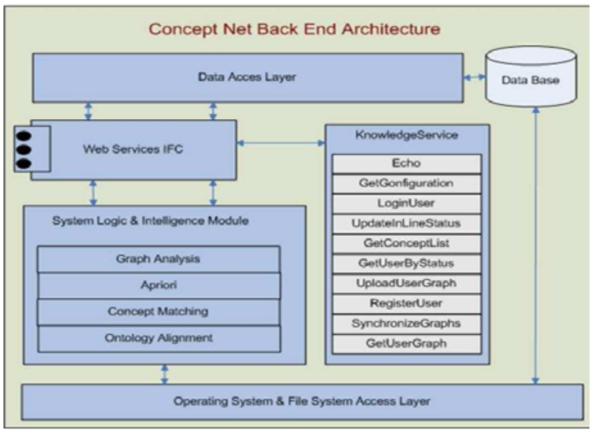

Fig. 3 The back-end module

The "System Logic & Intelligence Module" incorporates all the functionality dealing with the main operation of the proposed methodology. It includes submodules that implement the following functions:

Graph Analysis. This sub-module accepts an ontology in the form of graph and performs operations to analyze it in terms of paths, nodes, leaf nodes, parent and children nodes of every node. This functionality is achieved byusing the HP Jena API for ontology querying, in RDF/RDFs graph formats.

Apriori: Association Rule Mining Algorithm. This sub-module accepts as input all the possible paths and outputs the paths with the most significant associations among their nodes.

Concept Matching. This sub module inputs two sets of paths, outputs the most "interesting" ones to match each other.

Ontology Alignment. Based on the above, this sub-module performs the actual implementation and integration of the concepts into the user's graph.

The "Knowledge Service" component implements a set of function calls. When it is called properly from the "Web Services Interfaces" component, it performs the respective task. The function calls implemented in "Knowledge Service" are: Echo, GetConfiguration, LoginUser, UpdateInLineStatus, GetConceptList, GetUserByStatus, UploadUserGraph, RegisterUser, SynchronizeGraphs and GetUserGraph.

The "Database" stores all the system information, users' knowledge profiles, concepts, etc. Finally, the "Operating System & File System Access Layer" implements functionality for accessing the Operating System, so that the overall system can be easily transferred to a variety of Operating Systems.

The Concept Net system has been implemented for proof of concept purposes, as a full working prototype. It is applied for evaluation purposed to different domains. For instance, it is applied to the tourism domain [48]. The objective was to recognize among two Concept Net users (tourists), identify their personal ontology graphs, and analyze them by using the proposed methodology and finally exchange the concepts that interest each user.

IV. CONCLUSIONS

We present a system for detecting and then exchanging interesting parts of ontologies. To detect the interesting parts of ontologies, we define and evaluate a number of different measures of interestingness of parts of ontologies, in which each one represents different semantics of interestingness.

To exchange the interesting parts of ontologies, we propose a new ontology mapping technique, which exploits a structural similarity measure. Since it is based on the structure of ontologies, it can handle both metadata and instance heterogeneity. Moreover, it can easily be included in systems based on combination of mapping techniques, especially when only a few techniques are there for structural similarity ([13,16,22]). Also, it exhibits a low time complexity with respect to the related approaches.

ACKNOWLEDGMENT

We wish to thank Costas Sidiropoulos for his help during the tests.

REFERENCES

[1] R. Agrawal, H. Mannila, R. Srikant and A.I. Verkamo, Fast Discovery of Association Rules, in Advances in Knowledge Discovery and Data Mining, U.M. Fayyad, G. Piatetsky-Shapiro and P. Smyth Eds. AAAI Press/MIT Press, pp. 307, 1996.

[2] D. Aumueller, H.H. Do, S. Massmann, and E. Rahm, "Schema and ontology matching with COMA++", in Proc. SIGMOD (Demonstration), 2005.

[3] A. Balmin, V. Hristidis, and Y. Papakonstantinou, "Objectrank: Authority-based keyword search in databases", in Proc. VLDB, 2004, p. 564.

[4] T. Berners-Lee, "Weaving the Web", (Harper, 1999).

[5] J. Bock, R. Topor, and R. Volz, "Ontology Merging using Answer Set Programming and Linguistic Knowledge", in Proc. OAEI, 2007, p. 316.

[6] B. Boutsinas and T. Papastergiou, "On clustering tree structured data with categorical nature", Pattern Recognition, 41, pp. 3613-3623, 2008.

[7] S. Brin, R. Motwani, J.D. Ullman, and S. Tsur, "Dynamic Itemset Counting and Implication Rules for Market Basket Data", in Proc. ACM SIGMOD Int. Conf. Management of Data, 1997, p. 255.

[8] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine", Computer Networks, 30(1-7), pp. 107-117, 1998.

[9] H. Chalupksy, "OntoMorph: a translation system for symbolic knowledge", in Proc. 7th International Conference on Principles of Knowledge Representation and Reasoning, 2000.

[10] E. Compatangelo and H. Meisel, "Reasonable support to knowledge sharing through schema analysis and articulation"

[11] L. Ding, R. Pan, T. Finin, A. Joshi, Y. Peng and P. Kolari, Finding and Ranking Knowledge on the Semantic Web, ser. Lecture Notes in Computer Science, p. 156, (2005, vol. 3729.

[12] A. Doan, J. Madhavan, P Domingos, and A. Halevy, "Learning to map between ontologies on the semantic web", in Proc. 11th International World Wide Web Conference, 2002.

[13] M. Ehrig and S. Staab, "QOM Quick Ontology Mapping"

[14] M. Ehrig and Y. Sure, "Ontology Mapping - An Integrated Approach"

[15] T. d'Entremont and M.-A. Storey, "Using a degree-of-interest model for adaptive visualizations in Protégé", in Proc. 9th International Protégé Conference, 2006.

[16] J. Euzenat and P. Valtchev, "Similarity-based Ontology Alignment in OWL-Lite", in Proc. ECAI, 2004, p. 333.

[17] A. Firat and S. Madnick, "Knowledge Integration to Ontological Heterogeneity: Challenges from Financial Information Systems", in Proc. 23rd International Conference on Information Systems, 2001.

[18] F. Giunchiglia, P. Shvaiko and M. Yatskevich, "S-Match: an Algorithm and an Implementation of Semantic Matching", in Proc. ESWS, 2004.

[19] F. Giunchiglia, M. Yatskevich, P. Avesani and P. Shvaiko, "A Large Scale Dataset for the Evaluation of Ontology Matching Systems", Knowledge Engineering Review Journal, (to appear), 2008.

[20] E. Hovy, "Combining and Standardizing Large-Scale, Practical Ontologies for Machine Translation and Other Uses", in Proc. 1st International Conference on Language Resources and Evaluation, 1998.

[21] W. Hu and Y. Qu, "Falcon-AO: A practical ontology matching system"

[22] W. Hu, N. Jian, Y. Qu and Y. Wang, "GMO: a graph matching for ontologies", in Proc. K-CAPWorkshop on Integrating Ontologies, 2005, p. 41.

[23] M.N. Huhns and M.P. Singh, "Ontologies for agents", IEEE Internet Computing, 1(6), pp. 81-83, 1997.

[24] Y. Kalfoglou and M. Schorlemmer, Information-flow-based ontology mapping, ser. Lecture Notes in Computer Science, p. 1132, 2002, vol. 2519.

[25] Y. Kalfoglou and M. Schorlemmer, "Ontology Mapping: the State of the Art", The Knowledge Engineering Review, 18(1), 2003.

[26] S. Kaza and H. Chen, "Evaluating Ontology Mapping Techniques: An Experiment in Public Safety Information Sharing"

[27] C.-C. Kiu and C.S. Lee, "Ontology Mapping and Merging through OntoDNA for Learning Object Reusability", Educational Technology & Society, 9(3), pp. 27-42, 2006.

[28] M.S. Lacher and G. Groh, "Facilitating the exchange of explicit knowledge through ontology mappings", in Proc. 14th International FLAIRS Conference, 2001.

[29] A. Maedche and S. Staab, "Semi-automatic engineering of ontologies from texts", in Proc. 12th International Conference on Software Engineering and Knowledge Engineering, 2000, p. 231.

[30] A. Maedche, B. Motik, N. Silva and R. Volz, "MAFRA A MApping FRAmework for Distributed Ontologies"

[31] D.L. McGuinness, R. Fikes, J. Rice and S. Wilder, “An Environment for Merging and Testing Large Ontologies”, in Proc. 7th International Conference on Principles of Knowledge, Representation and Reasoning, 2000, p. 483.

[32] P. Mitra and G. Wiederhold, “Resolving Terminological Heterogeneity In Ontologies”, in Proc. ECAI02 workshop on Ontologies and Semantic Interoperability, 2002.

[33] N.F. Noy and M. Musen, “SMART: automated support for ontology merging and alignment”, in Proc. 12th Workshop on Knowledge Acquisition, Modelling and Management, 1999.

[34] N.F. Noy and M. Musen, “PROMPT: algorithm and tool for automated ontology merging and alignment”, in Proc. 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence, 2000, p. 450.

[35] N.F. Noy and M. Musen, “PROMPTDIFF: a fixed-point algorithm for comparing ontology versions”, in Proc. 18th National Conference on Artificial Intelligence, 2002, p. 744.

[36] N.F. Noy and M. Klein, “Ontology evolution: not the same as schema evolution”, Knowledge and Information Systems, 6(4), pp. 428-440, 2004.

[37] B. Omelayenko, “Integration of product ontologies for B2B marketplaces: a preview”, SIGecom Exch., 2(1), 2001.

[38] J S. Park, M. S. Chen and P. S. Yu, “An Effective Hash-Based Algorithm for Mining association rules”, in Proc. ACM SIGMOD Int. Conf. Management of Data, 1995, p. 175.

[39] S. Prasad, Y. Peng and T. Finin, “Using Explicit Information To Map Between Two Ontologies”

[40] A. Pretschner and S. Gauch, “Ontology based personalized search”, in Proc. 11th IEEE Intl. Conf. on Tools with Artificial Intelligence, 1999, p. 391.

[41] E. Rahm, and P.A. Bernstein, “A Survey of Approaches to Automatic Schema Matching”, VLDB Journal, 10(4), 2001.

[42] M. Sabou, V. Lopez and E. Motta, Ontology selection for the real semantic web: How to cover the queens birthday dinner?, in ser. Lecture Notes in Computer Science, p. 96, 2006.

[43] I. Schmitt and G. Saake, “Merging inheritance hierarchies for database integration”, in Proc. CoopIS98, 1998, p. 322.

[44] V. Spiliopoulos, A.G. Valarakos, G.A. Vouros and V. Karkaletsis, “Learning Subsumption Relations with CSR: A Classification based Method for the Alignment of Ontologies”, in Proc. OAEI, 2007, p. 321.

[45] G. Stumme and A. Maedche, “Ontology Merging for Federated Ontologies on the Semantic Web”, in Proc. International Workshop for Foundations of Models for Information Integration, 2001.

[46] C. Tatsiopoulos and B. Boutsinas, “Ontology Mapping based on association rule mining”, in Proc. 11th International Conference on Enterprise Information Systems, vol.3, 2009, p. 33.

[47] C. Tatsiopoulos, B. Boutsinas and K. Sidiropoulos, “On Aligning Interesting Parts of Ontologies”, in Proc. International Joint Conference on Knowledge Engineering and Ontology Development, 2009, p. 363.

[48] C. Tatsiopoulos and B. Boutsinas, “Automatic knowledge exchanging between tourists via mobile devices”, Journal of Hospitality and Tourism Technology, 1(2), pp. 163-173, 2010.

[49] Virtual Vineyards

[50] The W3 website. [Online]. Available: http://www.w3.org/DesignIssues/Principles.html.

[51] H. Wache, T. VÄogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann and S. HÄubner, “Ontology-Based Integration of Information - A Survey of Existing Approaches”, in Proc. IJCAI Workshop: Ontologies and Information Sharing, 2001.

[52] G. Wu, J. Li, L. Feng and K. Wang, “Identifying Potentially Important Concepts and Relations in an Ontology”, in Proc. International Semantic Web Conference, 2008, p. 33.