Crowdsourcing-Based Multi-Layer Automated Ontology Matching

An approach and Case Study

Alexander Smirnov, Nikolay Shilov, Nikolay Teslya
Laboratory of Computer Aided Integrated Systems
SPIIRAS
St.Petersburg, Russia
International Laboratory «Intelligent Technologies for Socio-Cyberphysical Systems»
ITMO University
St.Petrsburg, Russia
e-mail: {smir, alexey, nick, teslya}@iias.spb.su

Alexey Kashevnik
Laboratory of Computer Aided Integrated Systems
SPIIRAS
St.Petersburg, Russia
Department of Computer Science
Petrozavodsk State University (PetrSU)
Petrozavodsk, Russia
alexey@iias.spb.su

***Abstract*—This paper presents an approach and a case study for a multi-layer automated ontology matching based on the crowdsourcing technique. The main idea of our approach is that ontology matching is implemented automatically at first. In the case of non-adequate matching, the crowdsourcing technique is invoked, that involves crowd participants into the matching process. As a case study, the scenario of robot interaction is considered. The developed ontology matching approach allows providing for semantic interoperability between the robots for joint tasks solving.**

Keywords-ontology; ontology-matching; crowdsourcing; robots; interoperability.

I. INTRODUCTION

Ontology matching plays an important role in the development of ontology-based information systems. If a system consists of several interacting components and each component is developed by different manufactures based on different ontologies, we need to implement a matching of ontology entities between these components in order to provide semantic interoperability between them.

To implement such systems, the smart space technology can be used, which allows information sharing between different services of the system. This technology [1][2] aims at the seamless integration of different devices by developing ubiquitous computing environments, where different services can share information with each other, make different computations and interact for joint task solving. The open source Smart-M3 platform [3] has been used for the organization of the smart space infrastructure of the robots self-organization case study presented in the paper. The use of Smart-M3 platform enables significant simplification of further system development, including new information sources and services, making the system highly scalable. The key idea of this platform is that the formed smart space is device-, domain-, and vendor-independent. Smart-M3 assumes that devices and software entities can publish their embedded information for other devices and software entities through simple shared information brokers. The Smart-M3 platform consists of two main parts: information agents and kernel [4]. The kernel consists of two elements: Semantic Information Broker (SIB) and information storage. Information agents are software entities, installed on mobile devices of smart space users and other devices hosting smart space services. These agents interact with SIB through the Smart Space Access Protocol (SSAP). The SIB is the access point for receiving the information to be stored, or retrieving the stored information. All this information is stored in the information storage as a graph that conforms with the rules of the Resource Description Framework (RDF). In accordance with these rules, all information is described by triples "Subject - Predicate - Object".

The paper presents a multi-level automated ontology matching approach based on the crowdsourcing technique. Matching of ontology elements is implemented via automatic procedures and then enhanced manually if needed.

In the presented case study, the proposed algorithm is used to provide for interoperability support for robots solving a joint task.

The rest of the paper is structured as follows. The state-of-the-art of ontology matching systems is presented in Section II. Section III describes the proposed approach to ontology matching. Section IV presents the case study implemented using the proposed approach. The results are summarized in the Conclusion section.

II. STATE-OF-THE-ART

In order to analyze the existing ontology matching techniques, an extensive state-of-the-art review has been done, which covered systems/approaches/projects related to ontology matching. Among them the following ones are worth mentioning: GLUE System [5][6], Falcon-AO [7], MLMA [8], Hovy [9], SKAT [10], ONION [11], Promt [12], H-Match [13], CTX-MATCH [14], SMART [15], Cupid [16], COMA [17], Similarity Flooding Algorithm [18], AgreementMaker [19], Pattern Based Approach [20], MinSMatch [21], OntoView [22], Chimaera [16], VITRUVIUS [23][24], SAMBO [25], Falcon [26], DSSim [27], RiMOM [28], ASMOV [29], Anchor-Flood [30]. The following systems are more interesting and, for this reason, we describe them in detail below.

The VITRUVIUS platform integrates multiple sensors and handles different sensor configurations, allowing applications to be installed dynamically and run concurrently. A benefit of the platform is that it provides capabilities for reuse and evolution of existing sensor networks and applications, and for extensions by adding new sensors and applications. The platform uses an ontology-based approach to achieve semantic interoperability between different components. The authors understand the ontology as a vehicle that unifies the data originating from different system components into a universal understanding [24]. Ontology mappers are used to translate the local ontology (local syntactic and structural representations) into the application ontology and vice versa. From the development perspective, by using the ontology mapper, a new component (e.g., a sensor driver) can be integrated into the platform easily. This does not require re-implementing the component; only the ontology mapper requires an updated specification of how local terminologies and structures that refer to the new sensor can be represented in terms of the application ontology. The mapping between the application ontology and the local ontology is performed by the mapper component, which also provides interfaces for the data communication and control (e.g., sensor configuration). The mapper is implemented as an Android service, which can quickly be implemented using a generic development pattern.

SAMBO is a system for matching and merging biomedical ontologies. It handles ontologies specified in OWL language and outputs *1:1* alignments between concepts and relations. The system uses various similarity-based matchers, including.

- Terminological: n-gram, edit distance, comparison of the lists of words of which the terms are composed. The results of these matchers are combined via a weighted sum with pre-defined weights.
- Structural, through an iterative algorithm that checks if two concepts occur in similar positions with respect to is-a or part-of hierarchies relative to already matched concepts, with the intuition that the concepts under consideration are likely to be similar as well.
- Background knowledge-based, using (i) a relationship between the matched entities in Unified Medical Language System (UMLS) and (ii) a corpus of knowledge collected from the published literature exploited through a naive Bayes classifier.

The results produced by these matchers are combined based on user-defined weights. Then, filtering based on thresholds is applied to come up with an alignment suggestion, which is further displayed to the user for feedback (approval, rejection or modification). Once matching has been accomplished, the system can merge the matched ontologies, compute the consequences, check the newly created ontology for consistency, etc.

Falcon is an automatic divide-and-conquer approach to ontology matching. It handles ontologies in RDFS and OWL. It has been designed with the goal of dealing with large ontologies. The approach operates in three phases.

- Partitioning ontologies
- Matching blocks.
- Discovering alignments.

The first phase starts with a structure-based partitioning to separate entities (classes and properties) of each ontology into a set of small clusters. Partitioning is based on structural proximities between classes and properties, e.g., how closely are the classes in the hierarchies of rdfs: subClassOf relations and on an extension of the Rock agglomerative clustering algorithm [31]. Then, it constructs blocks out of these clusters. In the second phase, the blocks from distinct ontologies are matched based on anchors (pairs of entities matched in advance), i.e., the more anchors are found between two blocks, the more similar the blocks are. In turn, the anchors are discovered by matching entities with the help of the I-SUB string comparison technique [32].

DSSim is an agent-based ontology matching framework. The system handles large-scale ontologies in OWL and SKOS (Simple Knowledge Organization System) and computes *1:1* alignments with equivalence and subsumption relations between concepts and properties. It uses the Dempster-Shafer [33] theory in the context of query answering. Specifically, each agent builds a belief for the correctness of a particular correspondence hypothesis. Then, these beliefs are combined into a single more coherent view in order to improve correspondence quality. The ontologies are initially partitioned into fragments. Each concept or property of a first ontology fragment is viewed as a query, which is expanded based on hypernyms from WordNet [34], viewed as background knowledge. These hypernyms are used as variables in the hypothesis to enhance the beliefs. The expanded concepts and properties are matched syntactically to the similar concepts and properties of the second ontology in order to identify a relevant graph fragment of the second ontology. Then, the query graph of the first ontology is matched against the relevant graph fragment of the second ontology. For that purpose, various terminological similarity measures are used, such as Monger-Elkan and Jaccard distances, which are combined using Dempster's rule. Similarities are viewed as different experts in the evidence theory and are used to assess quantitative similarity values (converted into belief mass functions) that populate the similarity matrices. The resulting correspondences are selected based on the highest belief function over the combined evidences. Eventual conflicts among beliefs are resolved by using a fuzzy voting approach equipped with four ad hoc if-then rules. The system does not have a dedicated user interface but uses that of the AQUA (An Ontology-Driven Question Answering System) able to handle natural language queries.

RiMOM is a dynamic multi-strategy ontology matching framework. It focuses on combining multiple matching strategies, through risk minimization of Bayesian decision and quantitatively estimates the similarity characteristics for each matching task. These characteristics are used for dynamically selecting and combining the multiple matching methods. Two basic matching methods are employed.

- Linguistic similarity (edit distance over entity labels, vector distance among comments and instances of entities).

- Structural similarity (a variation of Similarity Flooding [18] implemented as three similarity propagation strategies: concept-to-concept, property-to-property and concept-to-property).

In turn, the strategy selection uses label and structure similarity factors, obtained as a preprocessing of the ontologies to be matched, in order to determine what information should be employed in the matching process. Specifically, the strategy selection dynamically regulates the concrete feature selection for linguistic matching, the combination of weights for similarity combination, and the choice of the concrete similarity propagation strategy. After similarity propagation, the matching process concludes with alignment refinement and extraction of the final result.

Automatic Semantic Matching of Ontologies with Verification (ASMOV) is an automatic approach for ontology matching that targets information integration for bioinformatics. Overall, the approach can be summarized in two steps.

- Similarity calculation.
- Semantic verification.

It takes two OWL ontologies and an optional alignment as the input and returns an *n:m* alignment between ontology entities (classes and properties) as the output. In the first step, it uses lexical (string equality, a variation of Levenshtein distance), structural (weighted sum of the domain and range similarities) and extensional matchers to iteratively compute similarity measures between two ontologies, which are then aggregated into a single one as a weighted average. It also uses several sources of general and domain specific background knowledge, such as WordNet and UMLS, to provide more evidence for similarity computation. Then, it derives an alignment and checks it for inconsistency. Consistency checking is pattern based, i.e., that instead of using a complete solver, the system recognizes sets of correspondences that are proved to lead to an inconsistency. The semantic verification process examines five types of patterns, e.g., disjoint-subsumption contradiction, subsumption incompleteness. This matching process is repeated with the obtained alignment as input until no new correspondences are found.

AgreementMaker is a system composed of a wide range of automatic matchers, an extensible and modular architecture, a multi-purpose user interface, a set of evaluation strategies, and various manual, e.g., visual comparison, and semi-automatic features, e.g., user feedback. It has been designed to handle largescale ontologies based on the requirements coming from various domains, such as the geospatial and biomedical domains. The system handles ontologies in XML, RDFS, OWL and outputs *1:1*, *1:m*, *n:1*, *n:m* alignments. In general, the matching process is organized into two modules: similarity computation and alignment selection. The system combines matchers using three layers.

- The matchers of the first layer compare concept features, such as labels, comments, instances, which are represented as TF_IDF (TF — Term Frequency, IDF — Inverse Document Frequency) vectors used with a cosine similarity metric. Other string-based measures, e.g., “edit distance” or “substrings”.
- The second layer uses structural ontology properties and includes two matchers called descendants similarity inheritance (if two nodes are matched with high similarity, then the similarity between the descendants of those nodes should increase) and siblings similarity contribution (which uses the relationships between sibling concepts).

At the third layer, a linear weighted combination is computed over the results coming from the first two layers, whose results are further pruned based on thresholds and desired output cardinalities of the correspondences. The system has a sophisticated user interface deeply integrated with the evaluation of ontology alignment quality, being an integral part of the matching process, thus empowering users with more control over it.

III. PROPOSED ONTOLOGY MATCHING APPROACH

A. *General Description*

The below proposed approach allows matching of two ontologies for the interoperability purposes of appropriate services and is based on the ontology matching model illustrated in Figure 1. The approach takes into account that the matching procedure has to be done in three steps. The first two steps are performed “on-the-fly” and the third step is optional; it is performed on-demand if the matching results of first two steps are not satisfactory. When a service joins a smart space, it performs a matching of its own ontology with the smart space ontology. If all classes that characterize service capabilities and requirements are matched with smart space ontology, the third step is skipped. The proposed approach also includes a graph-based distance improvement model that allows to propagate similarity from matched elements to elements related to them.

The approach consists of the following steps:

1. Compare all elements between two ontologies and fill the matrix M using *similarity-based model*. The matrix M is of size *m* to *n*, where *m* is the number of elements in the first ontology and *n* is the number of elements in the second ontology. Each element of this matrix contains the degree of similarity between the string terms of two ontology elements using the fuzzy string comparison method.

2. Calculate *semantic distances*, using background knowledge, e.g., WordNet or Wiktionary [35] and fill the matrix M’. The matrix M’ is of size *m* to *n*, where *m* is the number of elements in the first ontology and *n* is the number of elements in the second ontology. Each element of this matrix represents the degree of similarity between two ontology elements.

3. Update values in matrix M, where each new value of elements of M is the maximum value of (M, M’).

4. Improve distance values in the matrix M using the *graph-based distance improvement model*.

5. (Optional) Use the crowdsourcing technique for comparison of ontologies.

As a result, the matrix M contains the degrees of similarity between ontology elements of the two services. This allows determining correspondences between elements by selecting higher than the threshold value degrees of similarities.

B. *Similarity-Based Model for Matching Ontology*

The similarity-based model for the ontology matching is presented in Figure 2. It contains a stemming procedure to normalize words, improved fuzzy string comparison procedure, and normalization procedure.

To improve the matching quality, the application of the stemming procedure is proposed. This operation makes it possible to identify ontology elements even if they are written in different forms. The following conversions can be done: “looking” → “look”, “device” → “devic”, “vertical” → “vertic”, and “horizontal” → “horizont”. This procedure is uniquely tuned for each supported language.

The basis of the string comparison algorithm is the well-known conventional algorithm that calculates occurrence of substrings from one string in the other string. However, this algorithm does not take into account the length of the second string. As a result, it was decided to introduce the comparison based on the above algorithm twice: FC_1 = FuzzyCompare(Element$_1$, Element$_2$) and FC_2 = FuzzyCompare(Element$_2$, Element$_1$). After that we calculate the result as an aggregation of the above results in accordance with the following formula:

$Re' = n*FC_1+(1-n)*FC_2$, where n is a weight, $n \in [0;1]$.

$n = 0.5$ sets the same weight to the both strings, $n = 0$ searches only Request within Class, and $n = 1$ searches only Class within Request. It is proposed to set $n = 0.5$. Since the similarity metrics are obtained by different techniques they have to be normalized.

C. *Model of searching semantic distances of ontology elements*

To measure semantic distances between ontology elements we use the machine readable dictionary extracted by direct access to Wiktionary and WordNet.

This dictionary includes:

1) a set of words defined in dictionary along with,
2) definitions given for each word,
3) a set of synonyms for each word, and
4) a set of associated words for each word.

Words associated with a word are considered as hyperlinked words occurring in the Dictionary definition given for this word.

The nodes of ontology are linked to nodes representing their synonyms and associated words as this is given in the machine-readable dictionary. The links between the nodes are labeled by the distance of relations specified between the concepts represented by these nodes in the machine-readable dictionary. Weight w of a relation specified between two ontology elements t_i and t_j is assigned as:

$$w = \begin{cases} 0{,}5 & -t_i, t_j \text{ are synonyms} \\ 0{,}3 & -t_i, t_j \text{ are associated words} \\ \infty & -t_i, t_j \text{ are the same word} \end{cases}$$

The values for the weights were evaluated based on the following principles:

1. Semantic distances between synonyms are assumed to be smaller than semantic distances between associated words;

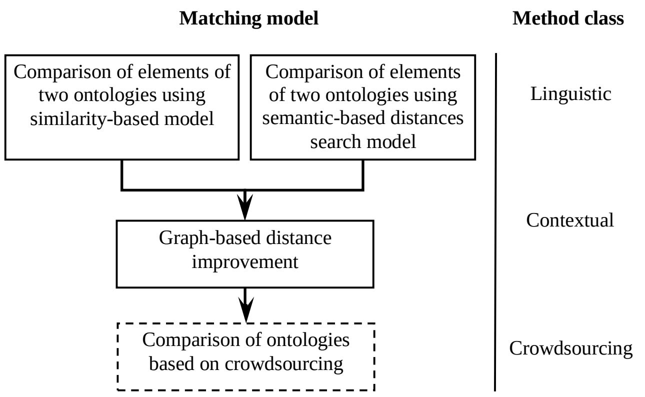
Matching model
Method class
Comparison of elements of two ontologies using similarity-based model
Comparison of elements of two ontologies using semantic-based distances search model
Graph-based distance improvement
Comparison of ontologies based on crowdsourcing
Linguistic
Contextual
Crowdsourcing

Figure 1. Multi-model approach to automated ontology matching.

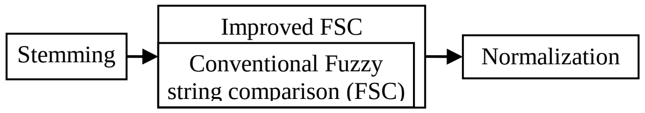

Figure 2. Similarity-based model

2. Semantic distance is proposed to be calculated as inversely proportional to weights raised to a power. The power is proportional to the path between the compared words. The longer the path, the greater the semantic distance for the two different words is expected to be. To meet this expectation with reference to the way of the semantic distance calculation, a weight of the relation between two different words should be in the range (0, 1) and ∞. Taken into account the first principle, we empirically selected the weights: 0,5 - for the relation between the synonyms; and 0,3 - for the relation between the associated words;

3. The semantic distance between the same words is equal to 0.

To search the semantic distance between the elements of two ontologies, the nodes of the first ontology are checked for their similarity to nodes of the second ontology. As a measure of similarity, the semantic distance (Dist) is used.

$$Dist(t_i, t_j) = \frac{1}{\sum_S \prod_{k=s_i}^{s_j} w_k}$$

where t_i, t_j – ontology elements; w – weight of lexical relation existing between t_i and t_j; S – a set of paths from t_i to t_j, where a path s is formed by any number of links that connect t_i and t_j passing through any number of nodes. The degree of similarity depends inversely on distance.

D. *Graph-based distance improvement model*

The graph-based improvement model for propagation similarities from one ontology element to another is presented in Figure 3 (see [36]). The main goal of this model is to propagate the degree of similarity between closely matching ontology elements to ontology elements related to them through RDF triples.

Let $X=(x_1, x_2, ..., x_n)$ be the set of subjects and objects in the ontology of two knowledge processors. Let $D_x = (d(x_i, x_j), ...)$ be a degree of similarity between x_i and x_j. Let $R = (r_1, r_2, ..., r_n)$ be a set of predicates in the ontology of two knowledge processors. Let $D_r = (d(r_i, r_j), ...)$ be a set of degrees of similarity between r_i and r_j. Constant Tr is a threshold value that determines whether two ontology elements mapped to each other or not.

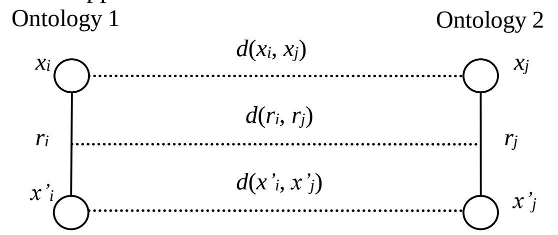

Figure 3. Matching of two ontology model

E. *Crowdsourcing comparison of ontologies*

Crowdsourcing is the process of obtaining information from an online group of crowd members. Technically, members are registered in a special application that provides them micro tasks and then summarizes their responses. For ontology matching process, this technique is used for making alignments between ontology elements if the methods presented above fail. If the service estimates that it needs more matching information it can use the crowdsourcing technique to improve the ontology matching with the help of a group of members.

Crowdsourcing system uses matrix M to show the ontology matching results found in steps 1–3 to the members, and provides a user friendly interfaces for them to see these alignments and to add, remove, or modify the matching between the ontology elements.

IV. CASE STUDY

The aim of the considered scenario is providing interoperability support for robots to solve a joint task. Two types of robots participate in the scenario: a pipeline robot and a manipulating robot (see Figure 4). The first one is stationary and has a pipeline that transfers objects from their current location to a predefined destination. It has a color sensor that determines the color of the transferred object. Robots interact in a smart space. To provide for semantic interoperability between robots, the proposed ontology matching approach is used. Each robot uploads its own ontology to the smart space when the robot joins it. The ontology matching service performs matching of the uploaded ontology with the smart space ontology and then extends the latter with the elements of the ontologies uploaded by robots.

The proposed interaction scheme of two robots in the smart space is presented in Figure 5. Example of robot interaction is shown in the example of sharing information about object transfer by the pipeline robot and color identification. When the pipeline robot is transferring the object, the pipeline velocity is shared with smart space by the following triple according to the pipeline robot ontology.

("Pipeline", "has_velocity", [pipeline velocity]).

When the color is determined, it is shared with smart space as follows.

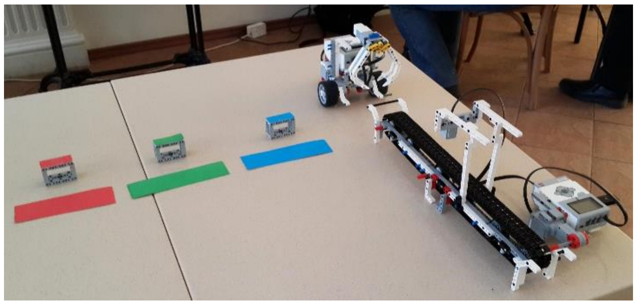

Figure 4. Pick-and-Place System Scenario

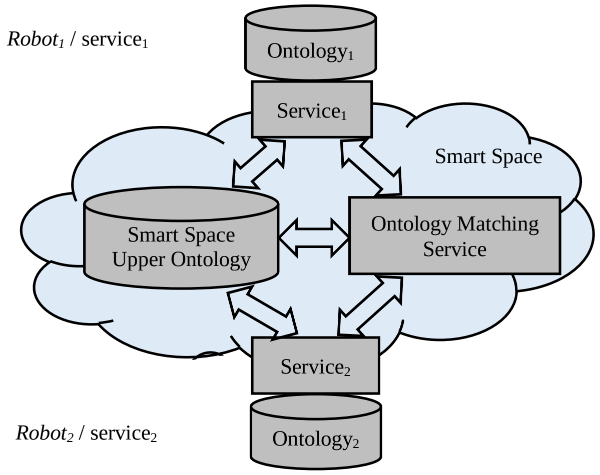

Figure 5. Robot Interaction in Smart Space Based on Ontology Matching

("Object", "has_color", [object color]).

When the object has been moved to the destination point and is ready for manipulation by the manipulating robot, the related triple is shared with smart space by pipeline robot.

("Object", "is_ready_for_manipulation", 1).

("Pipeline", "has_velocity", 0).

V. CONCLUSION

The paper presents the state-of-the-art of ontology matching works and proposes the crowdsourcing approach for matching ontology elements of a service-based system. The approach allows to implement matching of ontology elements with the help of group of people that improves the "on-the-fly" ontology matching approach. The considered case study is based on smart space technology that provides ontology-based information sharing between different system components and implements the ontology matching approach.

ACKNOWLEDGMENT

The presented results are part of the research carried out within the project funded by grants # 13-07-00336, 13-07-12095 of the Russian Foundation for Basic Research.

References

[1] D. J. Cook and S. K. Das, "How smart are our environments? an updated look at the state of the art", Pervasive and Mobile Computing, vol. 3, no. 2, pp. 53-73, 2007.

[2] S. Balandin and H. Waris, "Key properties in the development of smart spaces," Proc. 5th Int'l Conf. Universal Access in Human-Computer Interaction, Springer, 2009, pp. 3-12.

[3] Smart-M3 at Sourceforge, Web: http://sourceforge.net/projects/smart-m3 [retrieved: August, 2015].

[4] J. Honkola, H. Laine, R. Brown, and O. Tyrkko, "Smart-M3 Information Sharing Platform," Proc. ISCC 2010, IEEE Comp. Soc.; Jun. 2010, pp. 1041-1046.

[5] D. AnHai, M. Jayant, D. Pedro, and H. Alon, "Ontology Matching: A Machine Learning Approach," Handbook on Ontologies in Information Systems, 2004.

[6] A. Doan, J. Madhavan, P. Domingos, and A. Halevy, "Learning to map between ontologies on the semantic web," Proceedings of the 11th international conference on World Wide Web, 2002, pp. 662-673.

[7] W. Hu, N. Jian, Y. Qu, and Y. Wang, "GMO: A Graph Matching for Ontologies," K-CAP Workshop on Integrating Ontologies, 2005, pp. 43-50.

[8] A. Alasoud, V. Haarslev, and N. Shiri, "An Effective Ontology Matching Technique," 17th International Symposium ISMIS 2008, Toronto, Canada, May 2008, pp. 585-590.

[9] E. Hovy, "Combining and standardizing largescale, practical ontologies for machine translation and other uses," The First International Conference on Language Resources and Evaluation (LREC), Granada, Spain, 1998, pp. 535–542.

[10] P. Mitra, G. Wiederhold, and J. Jannink, "Semi-automatic Integration of Knowledge Sources," 2nd International Conference on Information Fusion, Sunnyvale, CA, July 1999.

[11] P. Mitra, M. Kersten, and G. Wiederhold, "Graph-Oriented Model for Articulation of Ontology Interdependencies," Proceedings of the 7th Int. Conf. on Extending Database Technology, Springer-Verlag, 2000.

[12] N. Noy and M. Musen, "Anchor-PROMPT: Using Non-Local Context for Semantic Matching," Workshop on Ontologies and Information Sharing at the Seventeenth International Joint Conference on Artificial Intelligence, Seattle, USA, 2001.

[13] S. Castano, A. Ferrara, and S. Montanelli, "H-Match: an Algorithm for Dynamically Matching Ontologies in Peer-based Systems," Proc. of the 1st VLDB Int. Workshop on Semantic Web and Databases, 2003.

[14] L. Serafini, P. Bouquet, B. Magnini, and S. Zanobini, "An algorithm for matching contextualized schemas via SAT," Technical report, DIT University of trento, Italy, 2003.

[15] N. Noy and M. Musen, "SMART: Automated Support for Ontology Merging and Alignment," 12th Workshop on Knowledge Acquisition, Modeling, and Management, Banff, Alberta, 1999.

[16] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder, "An Environment for Merging and Testing Large Ontologies," Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR'2000). Breckenridge, Colorado, USA, 2000, http://www.ksl.stanford.edu/software/chimaera/ [retrieved: August, 2015].

[17] D. Aumueller, H. Do, S. Massmann, and E. Rahm, "Schema and Ontology Matching with COMA++," Proceedings of the 2005 ACM SIGMOD international conference on Management of data, 2005, pp. 906-908.

[18] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity flooding: a versatile graph matching algorithm and its application to schema matching," Proceedings. 18th International Conference on Data Engineering, USA, 2002, pp. 117-128.

[19] I. Cruz, F. Antonelli, and C. Stroe, "Efficient Selection of Mappings and Automatic Quality-driven Combination of Matching Methods," The Fourth International Workshop on Ontology Matching, Washington DC., 2009.

[20] D. Ritze, C. Meilicke, O. Šváb-Zamazal, and H. Stuckenschmidt, "A pattern-based ontology matching approach for detecting complex correspondences," The Fourth International Workshop on Ontology Matching, Washington DC., 2009.

[21] [F. Giunchiglia, V. Maltese, and A. Autayeu, "Computing Minimal Mappings," The Fourth International Workshop on Ontology Matching, Washington DC., 2009.

[22] M. Klein, W. Kiryakov, D. Ognyanov, and D. Fensel, "Ontology Versioning and Change Detection on the We," 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02), Sigiienza, Spain, 2002.

[23] V. Bui, R. Verhoeven, and J. Lukkien, "A body sensor platform for concurrent applications," Proc. of IEEE Int. Conf. on Consumer Electronics, 2012, pp. 38-42.

[24] V. Bui, P. Brandt, H. Liu, T. Basten, and J. Lukkien, "Semantic Interoperability in Body Area Sensor Networks and Applications," 9th International Conference on Body Area Networks, London, Great Britain, Sep. 2014, pp. 210-216.

[25] P. Lambrix and H. Tan, "SAMBO – a system for aligning and merging biomedical ontologies," Journal of Web Semantics, vol. 4, no. 1, pp. 196-206, 2006.

[26] W. Hu, Y. Qu, and G. Cheng, "Matching large ontologies: A divide-and-conquer approach," Data and Knowledge Engineering, vol. 67, no. 1, pp. 140-160, 2008.

[27] M. Nagy and M. Vargas-Vera, "Towards an automatic semantic data integration: Multi-agent framework approach," Semantic Web, ch. 7, pp. 107-134, 2010.

[28] J. Li, J. Tang, Y. Li, and Q. Luo, "Rimom: A dynamic multistrategy ontology alignment framework," IEEE Transactoins on Knowledge and Data Engineering, vol. 21, no. 8, pp. 1218-1232, 2009.

[29] Y. R. Jean-Mary, E. P. Shironoshita, and M. R. Kabuka, "Ontology matching with semantic verification," Journal of Web Semantics, vol. 7, no. 3, pp. 235-251, 2009.

[30] M. S. Hanif and M. Aono, "An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size," Journal of Web Semantics, vol. 7, no. 4, pp. 344-356, 2009.

[31] S. Guha, R. Rastogi, and K. Shim, "Rock: A robust clustering algorithm for categorical attributes," Proc. 15th International Conference on Data Engineering, pp. 512–521, 1999.

[32] G. Stoilos, G. Stamou, and S. Kollias, "A string metric for ontology alignment," Proc. 4th International Semantic Web Conference (ISWC), 2005, pp. 624-637.

[33] G. Shafer, "A Mathematical Theory of Evidence," Princeton University Press, 1976.

[34] Wordnet, https://wordnet.princeton.edu/ [retrieved: August, 2015].

[35] Wiktionary, https://ru.wiktionary.org/ [retrieved: August, 2015].

[36] A. Smirnov, A. Kashevnik, N. Shilov, S. Balandin, I. Oliver, and S. Boldyrev, "On-the-Fly Ontology Matching in Smart Spaces: A Multi-Model Approach," Proceedings of the Third Conference on Smart Spaces, 2010, pp. 72-83.