

Aligning Smart and Control Entities in the IoT

Konstantinos Kotis, Artem Katasonov and Jarkko Leino

VTT, Technical Research Center of Finland

{Ext-konstantinos.kotis, artem.katasonov, jarkko.leino}@vtt.fi

Abstract. In our latest work, the problem of semantic interoperability for interconnected and semantically coordinated smart/control entities in a Semantic Web of Things has been explicated and a framework for Semantic Smart Gateways (SSGF) has been proposed. This paper aims to report on a proof-of-concept implementation of the core component of this framework which is the automatic alignment of smart/control entities' semantic descriptions. More specific, the paper reports on a recent approach towards implementing a configurable, multilingual and synthesis-based ontology alignment tool that has been evaluated by the SEALS and OAEI initiatives.

Keywords: ontology alignment, semantic coordination, smart entity, smart gateway, semantic interoperability

1 Introduction

One of the current trends in the Internet of Things (IoT) research domain is the integration of 'things' seamlessly with the existing Web infrastructure and the explosion of them uniformly as Web resources, shaping what is referred to as the Web of Things. Such an approach is a great facilitator of interoperability. However, for true interoperability within this setting, semantic interoperability is the key solver, i.e. the ability of the devices to unambiguously convey the meaning of data they communicate over Web protocols. Semantic Web technology can be used to extend WoT, shaping consequently what is sometimes referred to as the Semantic Web of Things.

The work reported in this paper is particularly motivated by our vision of an open and interoperable IoT, based also on the recent work of baseline solutions to cross-domain and cross-platform interoperability and information exchange approaches such as the Smart-M3. In this vision, the following four requirements must be satisfied:

- Ability to have gradually growing IoT environments, contrasted to installing and interconnecting all IoT devices and software at once.
- Ability to interconnect devices from different vendors.
- Ability of 3rd parties to develop software applications for IoT environments, contrasted to applications coming only from the devices' vendors.

adfa, p. 1, 2011.

© Springer-Verlag Berlin Heidelberg 2011

This work was carried out during the tenure of an ERCIM "Alain Bensoussan" Fellowship Programme. This Programme is supported by the Marie Curie Co-funding of Regional, National and International Programmes (COFUND) of the European Commission.

- Ability to develop applications that are generic in the sense of running on various IoT device sets (different vendors, same purpose), contrasted to developing applications for a very particular configuration of devices.

In contrast to the above vision, most IoT-related solutions today, except for UPnP/DLNA-based media sharing, fall into one of the following categories:

- Buy all the devices from one vendor.
- Connect “smart” devices (phones, TVs) from different vendors through installing a particular software client (from one vendor) on each of them (limited list of supported platforms).
- Use a particular gateway box, then can connect devices from different vendors (from a limited list of supported by the gateway).

In all three cases, a single vendor is responsible for all of the “interoperability” issues.

In our recent work on a Semantic Smart Gateway Framework (SSGF), we introduce a high-level type of entities, which we refer to as ‘smart entities’. They correspond directly to physical ‘things’ that are of some interest to humans, i.e. door, room, and package. In addition, they are equipped with other physical entities, the purpose of which is connecting the ‘things’ to IoT (e.g. sensors, actuators, RFIDs). Some smart entities, however, can be virtual and just equipped with datasets/data-streams exposed by Cloud services, e.g. Cosm (previously known as Pachube). Examples of smart entities are: a “smart door”, a “smart room”, a “smart package”. Smart entities may also be controlled by other entities responsible for the execution of tasks, which we refer to as ‘control entities’. More importantly, smart and control entities are equipped with a conceptual description (domain ontology) of the properties of the physical entities or functionalities they ‘carry’ and of the data they produce or consume.

The SSGF is designed to support the (semi-)automated translation process between smart or/and control entities’ data, with minimum human involvement, by computing their ontology definitions’ alignments. Specifically, the Semantic Smart Gateway Framework (SSGF) must provide (among other functionality), an ontology alignment component to support the discovery of similarities between smart/control entities’ profiles and ontology definitions, in order to be able to i) support the search for similar smart entities in a common view, ii) support the clustering of smart entities into domain-specific clusters, iii) to support the merging of similar smart entities towards a more efficient network organization, iv) to support the interoperability of smart entities with control entities (e.g. smart applications) that also carry ontology definitions of their input/output parameters. Such an approach is a great facilitator of interoperability and will be able to enable realization of the following example use cases:

- Two temperature sensors are both delivering measurements over HTTP GET as JSON, but of different structure and with different object/property names.
- Two heater devices are accepting commands over HTTP PUT as JSON, but of different structure and with different object/property names.
- A motion detector and light switch control software (one integrated application) is communicating messages (accepting measurements data and delivering commands in XML) with the related devices (communicating in JSON format) using heterogeneous vocabularies (e.g. app:Motion and dev:Movement). The application must be able to ‘understand’ motion detection events and issue commands to the switch

actuator e.g. for switching on/off the attached device (lamp). The switch actuator must be able to ‘understand’ commands issued by the application. The process must be automatic or at least semi-automatic with minimal human involvement.

- A package with an RFID or a UCODE label attached to it sent by a post office in origin-A can be automatically managed and forwarded by the intermediate destination-B to the next destination-C, without the current requirement that all post-offices (origin and destinations) share the same database or even the same semantic repository.

The paper is structured as follows: section 2 summarizes the most relevant and recent work, section 3 presents the aim, requirements, design and implementation of our approach, section 4 reports on the evaluation approach and results, and finally section 5 concludes the paper with plans for future work.

2 Related Work

This paper reports current research work related to a (semi-)automated ontology alignment process in the context of smart environments and IoT. This work extends the one of a dynamic ontology linking process for the behavioral coordination of heterogeneous systems that is reported in [2], and complements the related work on ontology mapping in Smart-M3 smart space that has been recently reported in [3]. The proposed work is also motivated by the related work of semantic coordination of agents in P2P systems [4, 5] in respect to the automatic and self-organization of schema/ontology mappings towards ‘healing’ incorrect mappings of entities in smart spaces. These works however are emphasizing the decentralized computation of mappings, a goal that is out of the scope of this paper. Moreover, our work is also motivated by the latest effort within the SWISS-Experiment project, emphasizing the use of W3C XG SSN ontology in a large-scale federated sensor network for semantic data sensor search [6], which manually provides mappings of data to the reference SSN ontology using a custom mapping language. Our work focuses on the automation of this manual mapping process.

Since the main problem of the presented work is the computation of alignments between ontologies, we mainly consider the related work that has been heavily conducted and reported the past six years within the Ontology Alignment Evaluation Initiative (OAEI) contests [7]. In this context, a number of tools have been evaluated and reported [7], and most importantly, a number of challenges have been identified [8]. The work presented in this paper is an extension to author’s work conducted for AUTOMS ontology alignment tool that has been participating in the OAEI 2006 contest, considering both these challenges and the specific requirements of the SSGF. Since it has been always acknowledged by OAEI contesters that there is not a single ‘best tool’ for ontology alignment [7, 8], and this has been driven due to the different application domains that tools must function, we do not compete with any of the related works. Instead, we focus on a comparative study based on their evaluation with common evaluation datasets.

3 Aligning Smart/Control Entities

In the SSGF, agents (software or human) must be able to register or search for smart or/and control entities (e.g. smart devices, smart applications) that match certain properties/criteria. Newly added heterogeneous entities, ‘carrying’ their vendors’ ontology definitions (or simpler types of metadata) for describing their properties, should be able to be dynamically coordinated via the automatic alignment of their ontology definitions. Such alignment may be computed in a direct (point-to-point) way or via agreed and shared conceptualizations of a reference ontology. In the latter case, in order to align the ontology definitions of two temperature sensors of different vendors, a reference representation of the temperature domain must exist.

To demonstrate the point-to-point alignment process, let us assume we have 2 sensors (a temperature and a motion detection), 2 actuators (switches) and 2 smart applications, with the following heterogeneous schemas (Class:{property TYPE}):

– Sensor: {movement_detection BOOLEAN, timed DATETIME}	(Sensor-A)
– Temperature_Sensor: {value DOUBLE, time_stamp DATETIME}	(Sensor-B)
– AirCond_SwitchAct: {value INTEGER, time_stamp DATETIME}	(Actuator-A)
– Lights_SwitchAct: {value INTEGER, time_stamp DATETIME}	(Actuator-B)
– MotionDetectionApp: {motionDetected BOOLEAN, timed DATETIME}	(Application-A)
– Temperatur: {wert DOUBLE, zeit_stempel DATETIME}	(Application-B, in German)

Application-A utilizes temperature data to issue commands towards a switch actuator for switching on/off the air-conditioning appliance, and application-B utilizes motion detection data to issue commands towards a switch actuator for switching on/off the lights in a room. The first goal is to match the sensor and actuator schemas with the proper application (input/output requirements) i.e. Sensor-A and Actuator-A with application-A, and consequently Sensor B and Actuator-B with application-B. The second goal is to align the elements (classes/properties) of each sensor schema with the elements of the application schema in order to be able for both sides (device and application side) to understand the semantics of each other. In the following paragraphs we focus on the implementation and evaluation of the second goal.

As abovementioned, in case smart/control entities are not equipped with an ontology, a transformation step (e.g. JSON to OWL representation) must also be added. An overview of the architecture of a “smart proxy” that implements this transformation step and the ontology alignment of two individual entities (in this case, of a smart entity and of a control entity) is depicted in Figure 1. An Ontology Wizard component is responsible for transforming device’s and application’s messages that are exchanged between each other or via a gateway (e.g. ThereGate) from JSON or XML or URI format to sets of OWL classes and properties as well as to refine those sets using some heuristic rules (e.g. to handle structural issues). The two sets of ontology elements, one for the device and one for the application, are then processed by the Ontology Alignment component in order to obtain their similarities and compute alignments between them. These alignments (computed at the deployment time) are then

used by a Message Translator component at run-time for a bi-directional translation of messages.

There are still some open issues that need to be treated in future work, i.e.:

- Where and how to place human involvement for the refinement of ontology definitions that cannot be fully automatically shaped from raw data (see Figure 1, ‘domain expert’ labeled grey face icon)
- Where and how to place human involvement (or other means of detecting and correcting erroneous alignment decisions e.g. wisdom of network) for the validation of the computed alignments between ontology definitions since a 100% accuracy of a fully automated method is not realistic (see Figure 1, ‘validator’ labeled grey face icon)

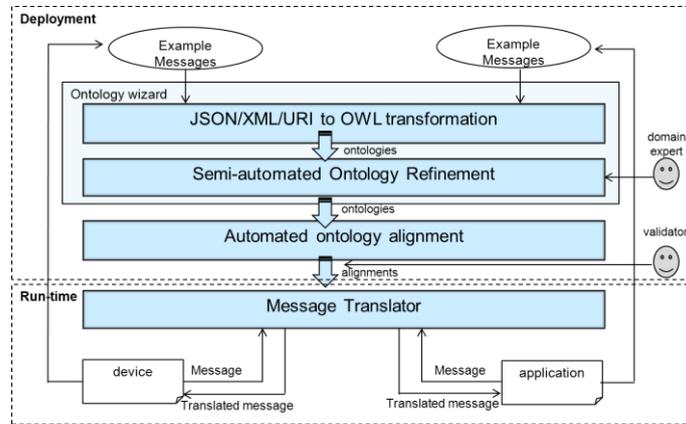


Fig. 1. ‘Smart proxy’ architecture for the (semi-)automated alignment of smart/control entities, providing syntactic and semantic interoperability of exchanged data between entities (in this case, device and application)

3.1 Requirements and design

Based on OAEI reports and best practices [7], we have observed that ontology alignment methods which utilize reference ontologies or other external resources such as WordNet lexicon, produce more precise results than others. Although to depend on a reference ontology and external resources is not (generally) a very good practice (must ensure that they both exist and provide a quite complete coverage of the domain knowledge), we observed that in our experimental setting seems to have more benefits than obstacles, thus we decided to use WordNet-based ontology alignment methods where possible.

Regarding the size of the ontologies that may be developed for representing the smart/control entities, it has been also observed that in most cases they will not exceed the number of 1 class and a couple of properties, with the rare case to reach the size of 3 or 4 classes. Such observations have been made by experimenting with

commercial products i.e. gateways and sensors-actuators for smart home (ThereGate¹) and smart building (Niagara Framework²). Small sizes of ontologies for smart/control entities allow applying a wide range of alignment methods, even those that have high computational complexity. The approach we follow in this work is able to identify the size of ontologies and adapt the alignment process accordingly by selecting the most suitable alignment methods (a process called profiling), as explained in the following sections.

The problem of computing alignments between ontologies can be formally described as follows: Given two ontologies $O_1 = (S_1, A_1)$, $O_2 = (S_2, A_2)$ (where S_i denotes the signature and A_i the set of axioms that specify the intended meaning of terms in S_i) and an element (class or property) E_i^1 in the signature S_1 of O_1 , locate a corresponding element E_j^2 in S_2 , such that a mapping relation (E_i^1, E_j^2, r) holds between them. r can be any relation such as the equivalence (\equiv) or the subsumption (\sqsubseteq) axiom or any other semantic relation e.g. meronym. For any such correspondence a mapping method may relate a value γ that represents the preference to relating E_i^1 with E_j^2 via r . If there is not such a preference, we assume that the method equally prefers any such assessed relation for the element E_i^1 . The correspondence is denoted by $(E_i^1, E_j^2, r, \gamma)$. The set of computed mapping relations produces the mapping function $f: S_1 \rightarrow S_2$ that must preserve the semantics of representation: i.e. all models of axioms A_2 must be models of the translated A_1 axioms: i.e. $A_2 \models f(A_1)$.

The synthesis of alignment methods that exploit different types of information (lexical, structural, semantic) and may discover different types of relations between elements has been already proved to be of great benefit [7, 8]. Based on the analysis of the characteristics of the input ontology definitions, i.e. the profiling of ontologies, our approach provides different configurations of alignment methods. The analysis of input ontologies is currently based on their size, the existence of individuals or not, the existence of class/properties annotations e.g. labels, and the existence of class names with an entry in WordNet lexicon. Part of the profiling is also a translation method that supports the translation of classes/properties annotations if these are given in a non-English language.

In the presented work we follow an advanced synthesis strategy, which performs composition of results at different levels (see Figure 2): the resulted alignments of individual methods are combined using specific operators, e.g. by taking the union or intersection of results, intersection of results or by combining the methods' different confidence values with weighing schemas. Given a set of k alignment methods (e.g. string-based, vector-based), each method computes different confidence values concerning any assessed relation (E_1, E_2, r) . The synthesis of these k methods aims to compute an alignment of the input ontologies, with respect to the confidence values of the individual methods. Trimming of the resulted correspondences in terms of a threshold confidence value is then performed for optimization.

The alignment strategy followed in our work is outlined in the following steps:

¹ <http://therecorporation.com/>

² <http://www.tridium.com/>

- Step 0: Analyze ontology definitions to be aligned (profiling) and assign the correspondent configuration of alignment methods to be used (configuration). If needed, translate ontology into an English-language copy of it.
- Step 1: For each integrated alignment method k compute correspondence $(E_i^1, E_j^2, r, \gamma)$ between elements of a peer ontology and a reference domain ontology definition that is specified in the IoT-ontology.
- Step 2: Apply synthesis of methods at different levels (using different aggregation operators) to the resulted set of alignments S_k .
- Step 3: Apply trimming process by allowing agents to change a variable threshold value for each alignments set S_k or for the alignments of a synthesized method.
- Step 4: Validate alignments (initially by human agents via ontology alignments' visualization interfaces) and allow for modifications.

The proposed ontology alignment approach considers most of the challenges in ontology alignment research [8, 9]) but emphasizes the a) alignment methods selection and synthesis, and b) user involvement. A general description of the ontology alignment process [8] extended with the profiling and configuration functionality is shown in Figure 2. Two alignment methods m and m' , also called matchers, are selected based on the profiling and configuration method and used for aligning the input ontologies o and o' . In case of translation needed, this is performed before entering m and m' respectively. An initial alignment A is often used also as input to the alignment methods but in our case this is always empty. The resulting alignments are aggregated/merged in (a) using an aggregation operator (union is the most common one), resulting in another alignment (A''') which will be improved by another alignment method (m'') resulting to the final alignment (A'''').

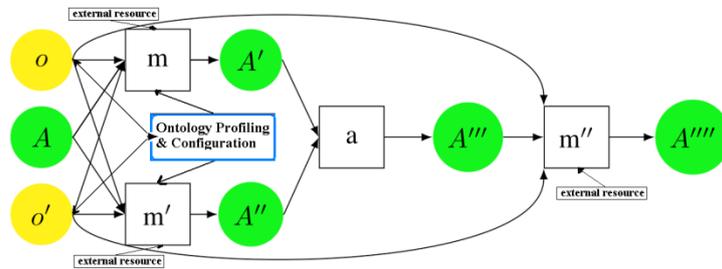


Fig. 2. General description of the ontology alignment process, extended with the profiling and configuration functionality

3.2 Implementation

AUTOMSv2 is an ontology alignment tool based on its first version in 2006 [10]. It computes 1:1 (one to one) alignments of two input domain ontologies, discovering equivalences between ontology elements, both classes and properties. The features that this new version integrates are summarized in the following points:

- It is implemented with the widely used open source Java Alignment API [11]

- It synthesizes alignment methods at various levels (lexical, structural, instance-based, vector-based, lexicon-based) with the possibility to aggregate their alignments using different aggregation operators (Union, Pythagorean means)
- It implements an alignment-methods' configuration strategy based on ontology profiling information (size, features, etc.)
- It integrates state-of-the-art alignment methods with standard Alignment API methods
- Implements a language translation method for non-English ontology elements

The tool has been developed from scratch, reusing some of the alignment methods already provided within the Alignment API. Other state-of-the-art methods such as the COCLU string-based and the LSA vector-based methods implemented in AUTOMS [10] using the AUTOMS-F API [12] have been re-implemented using the new API. The instance-based and structure-based alignment methods have been also implemented from scratch. The final experimental version of AUTOMSv2 (<http://ai-lab-webserver.aegean.gr/kotis/AUTOMSv2>) was delivered in 18th of March as a submission to the Ontology Alignment Evaluation Initiative 2011.5 Campaign (<http://oaei.ontologymatching.org/2011.5/seals-eval.html>), using the Semantic Evaluation At Large Scale (SEALS) platform. The detailed description of the alignment methods is out of the scope of this conference, hence of this paper also, since they have been adequately presented in previously published works [10, 12, 13].

The integrated string-based methods are used in two different synthesized methods and in one single method. All three methods use class and property names as input to their similarity distance metrics.

The first one synthesizes the alignments of two string-based similarity distance methods distributed with the Alignment API, namely, the 'smoaDistance' method and the 'levenshteinDistance'. A general Levenshtein distance between two strings is defined as the minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character. The one re-used from the Alignment API is a version of the general distance metric, based on the Needleman Wunsch distance method. The String Matching for Ontology Alignment (SMOA) method utilizes a specialized string metric for ontology alignment, first published in ISWC 2005 conference [13].

The second synthesized method synthesizes the alignments of two WordNet-based string-based similarity distance methods of the Alignment API, namely, the 'basicSynonymySimilarity' and the 'cosynonymySimilarity'. The first computes the similarity of two terms based in their synonymic similarity, i.e. if they are synonyms in WordNet lexicon (returns '1' if term-2 is a synonym of term-1, else returns a BasicStringDistance similarity score between term-1 and term-2), and the second computes the proportion of common synset between them, i.e. the proportion of common synonyms shared by both terms.

The third one is a single method that is implemented based on the state-of-the-art string similarity distance method COCLU, initially integrated in AUTOMS [10] and in other implementations using the AUTOMS-F API [12]. Since it is a complete re-implementation, in this version of the tool it is used in two different modes, i.e. in

names-mode and in labels-mode, according to the type of input ontologies that the profiling method will return. COCLU is a partition-based clustering algorithm which divides data into clusters and searches the space of possible clusters using a greedy heuristic.

Regarding vector-based alignment methods, AUTOMSv2 integrates two LSA-based methods, versions of the original HCONE-merge alignment method implemented in AUTOMS [10]. The first version is based on LSA (Latent Semantic Analysis) and WordNet and the second just in LSA. In the first one, given two ontologies, the algorithm computes a morphism between each of these two ontologies and a “hidden intermediate” ontology. This morphism is computed by the Latent Semantic Indexing (LSI) technique and associates ontology concepts with WordNet senses. Latent Semantic Indexing (LSI) is a vector space technique originally proposed for information retrieval and indexing. It assumes that there is an underlying latent semantic space that it estimates by means of statistical techniques using an association matrix ($n \times m$) of term-document data (WordNet senses in this case). The second version of this method is based on the same idea but instead of exploiting WordNet senses it builds the term-document matrix from the concepts’ names/labels/comments and their vicinity (properties, direct superconcepts, direct subconcepts) of the input ontologies. The similarity between two vectors (each corresponding to class name and annotation as well as to its vicinity) is computed by means of the cosine similarity measure.

Finally, two more methods, a structure-based and an instance-based method, are integrated, based on the general principle that two classes can be considered similar if a percentage of their properties or their instances has been already considered to be similar. The similarity of properties and instances is computed using a simple string-matching method (Levenshtein). Although it seems that structure and instances are not present in the ontology definitions of smart/control entities, their integration in AUTOMSv2 does not influence its performance since, as already stated, the profiling analysis automatically detects the features of the input ontologies and exclude these methods from computing alignments (i.e. are not included in the synthesis configuration for the smart/control entities’ ontology definitions).

The different configurations regarding the way the above methods were synthesized, i.e. computing and merging alignments, is based on the profiling information gathered after the analysis of the input ontologies. Both input ontologies (since our problem concerns the alignment of two ontologies), are examined using currently four different analysis methods:

1. Based on the size of the ontologies, i.e. the number of classes that ontologies have, if one of them has more than a specific number of classes (this number is currently set to 100), then this pair of ontologies is not provided as input to alignment methods with heavy computations since it will compromise the overall execution time of the tool. Such methods are the vector-based, WordNet-based and structure-based ones.
2. If an ontology pair has an ontology with no instances at all, then this pair is not provided as input to any instance-based alignment method (the explanation for this is straight forward).

3. If an ontology pair has two ontologies that half of their classes have no names with an entry in the WordNet, but they have label annotation(s), then provide this pair as input to alignment methods that a) do not consider WordNet as an external resource and b) consider labels matching instead of class names.
4. If an ontology pair has two ontologies that half of their classes have no names with an entry in the WordNet, and they also have no labels, then provide this pair as input to alignment methods that a) do not consider WordNet as an external resource and b) do not consider labels matching. In fact, this is the hardest case, and we have applied an experimental configuration using either instance-based methods if instances exist, or COCLU alignment method for class names string matching.

AUTOMSV2 is re-using a free Java API named WebTranslator (<http://webtranslator.sourceforge.net/>) in order to solve the multi-language problem. AUTOMSV2 translation method is converting the labels of classes and properties that are found to be in a non-English language (any language that WebTranslator supports, including Chinese and Russian) and creates a copy of an English-labeled ontology file for each non-English ontology. This process is performed before AUTOMSV2 profiling, configuration and matching methods are executed, so their input will consider only English-labeled copies of ontologies.

4 Evaluation

We have evaluated our tool with OAEI 2011 and 2011.5 contests datasets, measuring precision and recall. Precision is the ratio between true positive and all aligned objects (the correct alignments among the retrieved alignments have to be determined); Recall is the ratio between the true positive and all the correspondences that should have been found.

We present and compare results using datasets of OAEI 2011 contest, since the latest results of 2011.5 contests were not published until the submission of this paper. The dataset, namely ‘Benchmarks2’, concerns new systematically generated benchmarks, based on other ontologies than the bibliographic one (Benchmark1). It is made of a set of 103 pairs of ontologies related to the conference publications domain. The Ekaw ontology, one of the ontologies from the conference track, has been used as reference ontology for generating the dataset. It contains 74 classes and 33 object properties. Figure 3 presents the mean results of precision (Prec) and recall (Rec) of all tools that have been evaluated with this dataset.

We have computed three different average types, i.e. *h-mean*, *mean* and *mean+* in order to better understand the performance of our tool. H-mean (harmonic mean of non-zero values) and mean+ (simple average of non-zero values) implement averages that do not consider zero values in the computations. Zero values occur only for some ontology pairs (15 out of the 103) that organizers have completely changed by scrambling all possible elements’ strings (names, labels, comments) using randomly generated meaningless strings. We conjecture that such ontologies are not much realistic and rare to find in real smart space environments.

As it is depicted in Figure 3, based on the different average computation types, AUTOMSV2 performs better regarding precision (from 0.83 to 1.00) than recall (from 0.49 to 0.62). Also, from the 16 tools compared, AUTOMSV2 outperforms 6 of them in terms of MEAN precision and in terms of H-Mean and Mean+ it performs with a near to 100% accuracy. Also, only 6 out of the 16 tools achieved a better Mean recall than AUTOMSV2 did, however it outperforms almost all other tools in terms of H-mean recall.

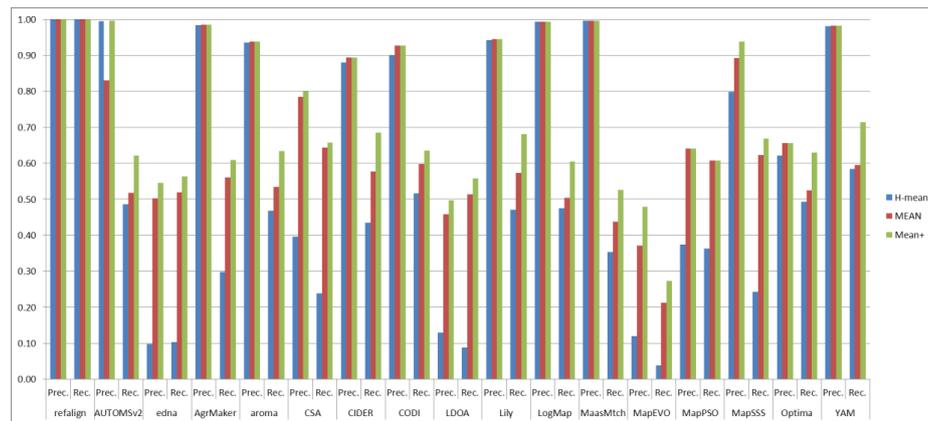


Fig. 3. OAEI 2011 Benchmark mean results compared to AUTOMSV2

It is our aim to create a new custom evaluation dataset that is focused on data gathered from the smart environments domain. The aim is to develop experimental OWL ontology definitions that represent such knowledge and evaluate our tool with ontology pairs that combine different variations of them.

5 Conclusion and Future Work

This paper presents an implementation of SSGF core component, i.e. the automatic alignment of smart/control entities' semantic descriptions. More specifically, it presents the implementation of AUTOMSV2 ontology alignment tool that has been designed in order to meet the alignment requirements of smart/control entities in the SSGF. The focus of the presented implementation is on a) the alignment-methods' configuration strategy based on ontology profiling information, the synthesis and integration of state-of-the-art ontology alignment methods, and c) the support of multilingualism. The presented tool is part of a proof-of-concept SSGF implementation that is called 'smart-proxy', for the (semi-)automated alignment of smart/control entities, providing syntactic and semantic interoperability of exchanged data between these entities. Future work concerns issues related to the degree and nature of human involvement during the ontology refinement and ontology alignments' validation process. Finally, the paper presents the overall performance of the implemented tool and compares it with other tools using the latest OAEI datasets.

Acknowledgements

Authors acknowledge that part of the work presented in this paper has been driven by related work in the following projects: Internet of Things (Finnish national TEKES project), iCORE (287708 - FP7 Integrated Project). We also thank the anonymous reviewers for their valuable comments.

References

1. Kotis, K., Katasonov, A.: Semantic Interoperability on the Web of Things: The Semantic Smart Gateway Framework. To appear in: Proceedings of the 6th International Workshop of Frontiers in Complex, Intelligent and Software Intensive Systems, Palermo, Italy (2012)
2. Katasonov, A., Terziyan, V.: Using Semantic Technology to Enable Behavioral Coordination of Heterogeneous Systems, In: Semantic Web, pp. 135-156 (2010)
3. Smirnov, A., Kashevnik, A., Shilov, S., Balandin, S., Oliver, I., Boldyrev, S.: On-the-Fly Ontology Matching in Smart Spaces: A Multi-model Approach, In: S. Balandin et al. (Eds.): ruSMART/NEW2AN 2010, LNCS 6294, pp. 72–83 (2010)
4. Cudré-Mauroux, P., Suchit, A., Karl, A.: GridVine: An Infrastructure for Peer Information Management and Large-Scale Collaboration, IEEE Internet Computing, 11(5), 36-44 (2007)
5. Spiliopoulos, V., Vouros, G. A.: Synthesizing Ontology Alignment Methods Using the Max-Sum Algorithm, Knowledge and Data Engineering, IEEE Transactions on, 24(5), 940-951 (2012)
6. Calbimonte, J., Jeung, H., Corcho, O., Aberer, K.: Semantic Sensor Data Search in a Large-Scale Federated Sensor Network, In: Semantic Sensor Network Workshop, ISWC, pp. 23-38, CEUR-WS proceedings <http://CEUR-WS.org> (2011)
7. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology Alignment Evaluation Initiative: six years of experience, J. Data Semantics 15: 158-192 (2011)
8. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges, IEEE Transactions on Knowledge and Data Engineering, 08 Dec. 2011. IEEE computer Society Digital Library. IEEE Computer Society, <<http://doi.ieeecomputersociety.org/10.1109/TKDE.2011.253>>
9. Kotis, K., Lanzemberger, M.: Ontology Matching: Current Status, Dilemmas and Future Challenges. In: International Conference of Complex, Intelligent and Software Intensive Systems, pp. 924-927 (2008)
10. Kotis, K., Valarakos, A., Vouros, G. A.: AUTOMS: Automating Ontology Mapping through Synthesis of Methods, In: International Semantic Web Conference, Ontology Matching International Workshop, Atlanta USA (2006)
11. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The Alignment API 4.0, Semantic Web - Interoperability, Usability, Applicability, 2(1):3-10, IOS Press (2011)
12. Valarakos, A., Spiliopoulos, V., Kotis K., Vouros, G. A.: AUTOMS-F: A Java Framework for Synthesizing Ontology Mapping Methods, In: International Conference i-Know, Graz, Austria (2007)
13. Stoilos, G., Stamou, G., Kollias, S.: A String Metric for Ontology Alignment. In: International Semantic Web Conference (2005)