Proposal for the combination of ontology assemble and ontology mapping processes

Nuno Silva, Jorge Santos and João Rocha
GECAD - Knowledge Engineering and Decision Support Group
Institute of Engineering - Polytechnic of Porto
Porto, Portugal
{nps,ajs,jsr}@isep.ipp.pt

***Abstract*—**

Ontology mapping is dependent on the matching algorithms, which in turn depends on the semantics of the ontologies. This proposal grounds on two distinct technologies: ontology mapping and ontology assemble. The ontology mapping system is a service-oriented system in which autonomous plug-able services capture and represent the ontology mapping domain's expertise. Automatic ontology mapping systems perform poorly due to the lack of ontologies' semantics to reason upon. The assemble process aims to integrate well-founded, proofed knowledge into the the domain ontologies, providing extra semantics. Exploiting such semantics in distinct phases of the ontology mapping process, we envisage that such semantics would be very useful in the improvement of the automatic mapping results. Conversely, the ontology mapping experiences would provide feed-back to the assemble process, suggesting its improvement too. The cyclic improvement would run indefinitely.

This paper suggests therefore the combination of efforts from both the assemble and mapping processes towards a better semantic relations and assembled ontologies.

Keywords:Knowledge Representation, Ontologies, Knowledge Management

I. Introduction

Semantic Web suggests the annotation of Web resources with machine processable meta-data, enabling computer devices to analyze the meaning and the semantic relations between documents and their parts. Ontologies as means for conceptualizing and structuring knowledge allows the explicit specification of a domain of discourse, permitting the access and reasoning about agent's knowledge. Ontologies raise the level of specification of knowledge by the combination of data and its semantics, while enabling its exchange in an explicit understandable form. Semantic Web and ontologies are therefore fully geared as a valuable framework for distinct applications requiring knowledge-based interoperability, like business applications (E-Commerce, B2B) and information retrieval.

These online services require fast and reliable interoperability processes between autonomous and semantically heterogenous entities. Protocols, messages and their contents are all potentially described and shared through ontologies. However, ontologies do not overcome per se any interoperability problem, since it is hardly conceivable that a single ontology is applied for all kind of domains and applications.

A. Ontology Mapping

Ontology mapping, as a reconciliation approach, is seen as a key technology for the knowledge-based interoperability in the semantic web and in other information distributed systems. Ontology mapping does not intend to unify ontologies and their data, but to transform ontology instances between intervenient entities according to the semantic relations defined at conceptual level (figure 1). Repositories are therefore kept separated, independent and distinct, maintaining their complete semantics and contents.

Though the ontology mapping process [21] is much more complex, in the context of this paper we focus only on the discovery and specification of the semantic relations. In the discovery (matching) phase of the process, similarities between entities of both ontologies are detected and measured, which are further exploited in the bridging phase, that specifies which entities are semantically related and what type of semantic relation existent between them. For each new ontology added to the interoperability context, a new ontology mapping process occurs.

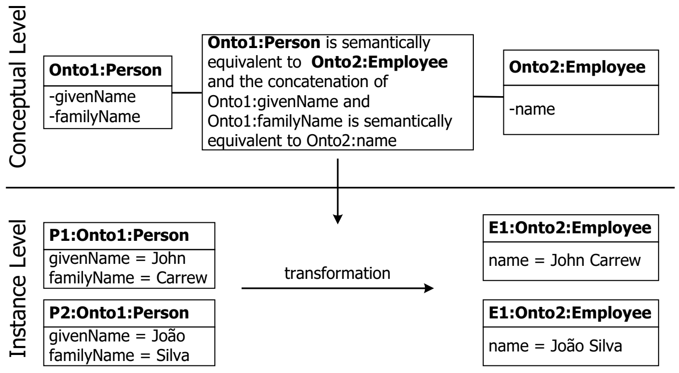

Fig. 1. Informal representation of ontology mapping

Ontology mapping is a time-consuming, human-intensive process, but unfortunately automation approaches are highly inefficient and error-prone, specially because ontologies do not enclose sufficient semantics to reason upon. In particular ontologies lack formal specification of contents.

B. Ontology Assembly

In recent years, we have seen a surge of ontologies and ontology technology with many ontologies now being available on the Web. At the same time one could observe that most ontologies (e.g., consider the DAML ontology library at `http://www.daml.org/ontologies/`) engineered exhibit only rather simple structures, *viz.* taxonomies and frame-like links between concepts.

This observation might indicate that such — comparatively — simple structures are sufficient for the large majority of ontology-based systems. According to our own experiences about ontologies for knowledge portals [24] and power systems [18], however, one frequently needs intricate concept descriptions and interactions — in particular ones about time and space.

Because of intense (and fruitfully ongoing) research, many practical theories about time and space are now well understood. While the same can be said about the engineering of concept hierarchies and concept frames, the issue of how to engineer complex ontologies with intricate interactions based on time has not been researched very deeply, yet, rendering the engineering of a new complex domain ontology with time and/or space is a labor intensive, one-off experience with little methodology.

Although strictly speaking orthogonality is a relation between ontologies, not a property of an ontology itself, in the scope of this paper *Orthogonal Ontology* stands for an ontology describing a category of knowledge that for its generic nature crosses multiple (application) domains, which can be further re-used (e.g., extended, specialized) in the development of domain and task ontologies. The *Orthogonal Ontology* may refer to very conceptual subjects like time, space, things, but is also expected to model specific knowledge (e.g., chemistry elements in the periodic table) commonly used in pharmacy or chemistry industry. Guarino [7] proposes ontology engineering based in well-founded theories like identity, *Mereology*, *Topology* and *Morphology*. As in many other areas, well-founded theories allows to systematize processes and encompass a modelling and V&V proofs leading to easier reutilization and integration.

However, the difficulty to model these issues, aggravated by the gap between the theory and implementation, lead to their limited specification. Low reusability and low quality and high maintenance costs of knowledge-based systems in general and of ontologies in particular, are further side-effects.

Factorization of problem and reutilization of (tested and validated) components are far-used in software engineering, towards better and lower-cost solutions. Ontology assemble process (figure 2) suggests the integration of formally, complex but common elements into domain and task ontologies.

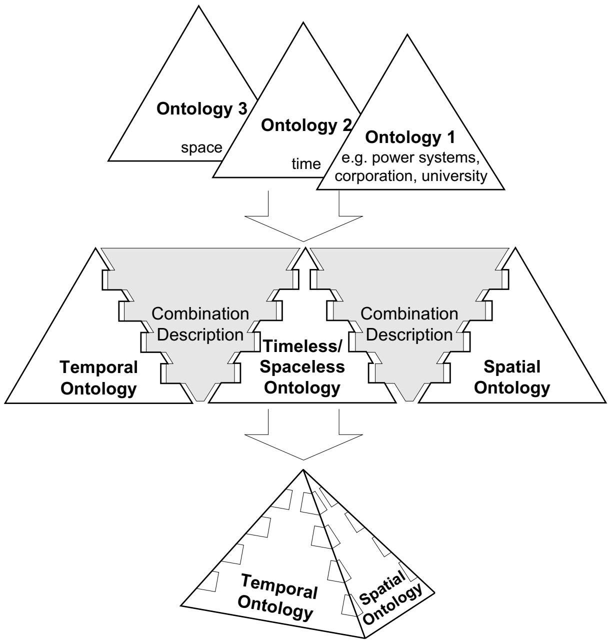

Fig. 2. Factorization for ontology engineering

C. Proposal

The integration of these foundational elements in a variety of domain systems provides accurate proofed semantic elements, upon which would be possible to draw better semantic relations.

This paper proposes the combination of efforts from both mapping and assembly processes, towards quality and usefulness of formal orthogonal elements. On one hand, mapping process would benefice from the more accurate and suited semantics of the elements. On the other hand, the assembly process would benefice from the requirements and improvements suggested by the mapping process.

The rest of the paper is organized as follows. After pointing the motivations for this work in the current section. We describe the foundations considered in the elaboration of this proposal, mapping and assembling of ontologies, as well as the tools developed to support these processes, MAFRA and FONTE, respectively. Then we present the proposal for combining efforts from both processes in a common framework. Finally, we relate to other work and conclude pointing some aspects to be considered in a future work.

II. Foundations

A. Mapping

This section describes the ontology mapping system architecture conceived according to the MAFRA - MApping FRAmework [21].

Besides the fundamental characteristics inherited from MAFRA, the system architecture benefices from two complementary observations:

1. It is virtually impossible to provide *a priori* all possible transformation requirements using a monolithic and static ontology mapping system;
2. The transformation results of and ontology mapping execution largely depends on the transformation service associated with semantic bridges.

The adoption of a modular, open architecture arises therefore as a natural approach (figure 3), in which independent transformation modules, referred as Services, are attached to the system functional core modules (i.e. bridging, execution, negotiation, evolution, etc.) [21]. This service-oriented approach advocates the need to exploit the knowledge and capabilities associated with each Service in order to increase automation and quality of the overall mapping process. Services represent expertise in applying transformation upon the ontology instances, but not only. In fact, it is suggested that Services expand their expertise to other ontology mapping process phases, in special to the matching phase. Evolution, validation and negotiation of of semantic relations are other phases benefiting from this service-oriented architecture.

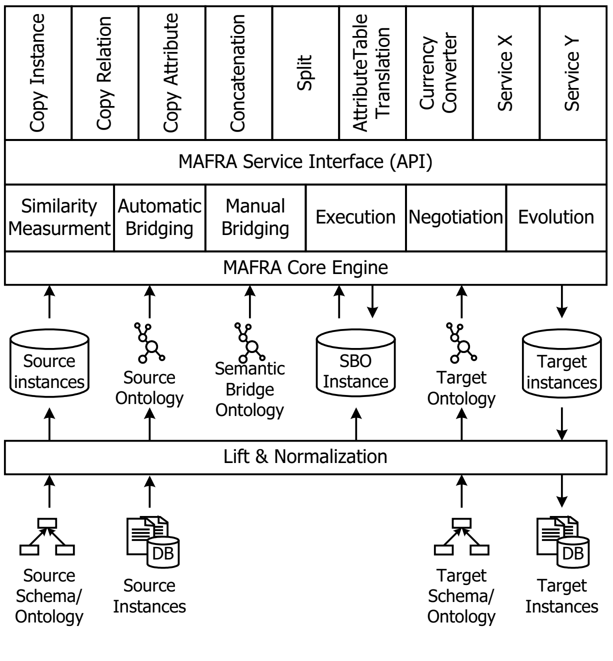

Fig. 3. Service-Oriented Achitecture

The more capabilities each service implements the more useful it is to the MAFRA Core Modules and to the ontology mapping system as a whole. Services are described and specified through the instantiation of a simple ontology, which, in the line of other approaches (e.g., OWL-S[1]) describes the characteristics of services, their input and output parameters, order, constraints, etc.

In particular, each service is able and autonomous to decide which and how similarity measures are combined in order to determine the validity of a semantic relation between a set of source and target ontologies' entities. For example, imagine that the Concatenation service exists in the system, which has the ability to concatenate source ontology's attributes into one target ontology attribute. This service would describe their matching requirements as (1) all entities should be string attributes, (2) at least two source entities should be related to one (and just one) target entity (n:1 cardinality) and (3) the name or lexicons associated with source entities should be hyponyms[2] of all the name or lexicons associated with the target entity. Other, more complex requirements are necessary, but their use depends on the capabilities of the system to exploit the ontologies' semantics in discovering true similarities. This is the task for matchers.

We envisage a modular, open architecture where new matching algorithms are dynamically plugged into the matching module according to availability and services requirements. Each algorithm, named Matcher, is expert in measuring a specific type of similarity between ontologies' entities, but are not focused on draw a conclusion on the ultimate similarity. This is a competence of the service. Accordingly, each service would seek similarities from multiple matchers.

Services are further able to determine if a set of source ontology's entities are semantically equivalent to a set of target ontology's entities according to the included expertise.

For each semantic equivalence drawn, an instance of a semantic relation is specified. These instances are referred as semantic bridges, and are the mapping elements who wrap all the information necessary to perform the instances' transformation. Semantic Bridging Ontology (SBO) is an ontology of semantic relations between two ontologies, that exploit the idea of service described above. Though no deep details are necessary concerning the concept of semantic bridge, more information can be found in [22].

B. Assembling

FONTE (Factorizing ONTology Engineering complexity) is a ontology engineering methodology that pursues a 'divide-and-conquer' strategy for engineering complex ontologies with time. FONTE divides a targeted ontology that is complex and includes time into two building blocks, a temporal theory and a time-less domain ontology. Each one of the two sub-ontologies can be built independently allowing for a factorization of complexity. The targeted ontology is then assembled from the time-less domain ontology and the temporal theory by the operator $\otimes$.

Thereby, the assembling operator $\otimes$ is very different from existing operators for merging or aligning ontologies [15, 17].

[1]http://www.daml.org/services

[2]*source entity* is a kind of *target entity*

Merging ontologies is a process that intends to join different ontologies about overlapping domains into a new one and most of its problems and techniques are related to the identification of similar concepts through structure analysis (e.g., graph analysis, path length, common nodes or/and edges and lexical analysis). For instance, car from ontology 1, *O1.car*, and auto from ontology 2 *O2.auto* may be defined to be identical in the process because of results of the structure analysis. To formalize the merging and aligning process, Wiederhold proposed a general algebra for composing large applications through merging ontologies of related domains [25] and actually, the operations proposed (*Intersection*, *Union* and *Difference*) are about the similarities and differences of two ontologies.

In contrast, the result of $\otimes$ needs rather to be seen in rough analogy to the Cartesian product of two entities. For instance, car from ontology 1, *O1.car*, with its frame *O1.licensedInState* is assembled by $\otimes$ with ontology 2 and its *O2.timeInterval* in a way such that every car in the result ontology has a lifetime as well as multiple *O1.licensedInState*-frames with different, mutually exclusive life spans.

The assembly process comprises two main building blocks. First, the specification of temporal aspects for a time-less domain ontology remains dependent on the conceptualization of the ontology engineer. In fact, the example used for illustrating the assembly of general axioms below shows that there are ontological decisions to be made that can not be derived from the structure analysis of the two ontologies and therefore require human interaction. Second, in order to facilitate and accelerate the assembly of time-less domain concepts with temporal notions, the interactive process is supported by heuristics asking and pointing the engineer.

The assembly process runs as depicted in figure 4: It starts by an Initial Setup. Some basic operations are performed, namely loading the ontologies to be assembled, loading a set of rules to drive the process and initializing some process parameters. The rules and parameters are defined separately from the tool in order to allow for adaptations to the particular needs of different temporal ontologies. However the rules and parameters do not change when a new domain ontology is to be assembled. The Target Ontology initially corresponds to the union of the time-less domain ontology, *O1*, and the temporal theory, *O2*.

Initially, the user may re-structure some part of the domain ontology to include temporal aspects by defining and executing (what we call) task instances. When performing such re-structuring task instances, a structure analysis finds possibly implicated task instances and proposes them onto the Task List. In subsequent iterations the engineer decides whether to accept an automatically proposed task instance from the Task List. Alternatively, the user may take new initiatives and define and execute a new task instance from scratch.

For manually defined task instances, a set of logical tests (Validate) are performed in order to detect the existence of any knowledge anomalies (e.g., circularity or redundancy [16]). In contrast, the acceptance of a proposed task instance does not require further checks as the checks are tested for validity before the user sees them.

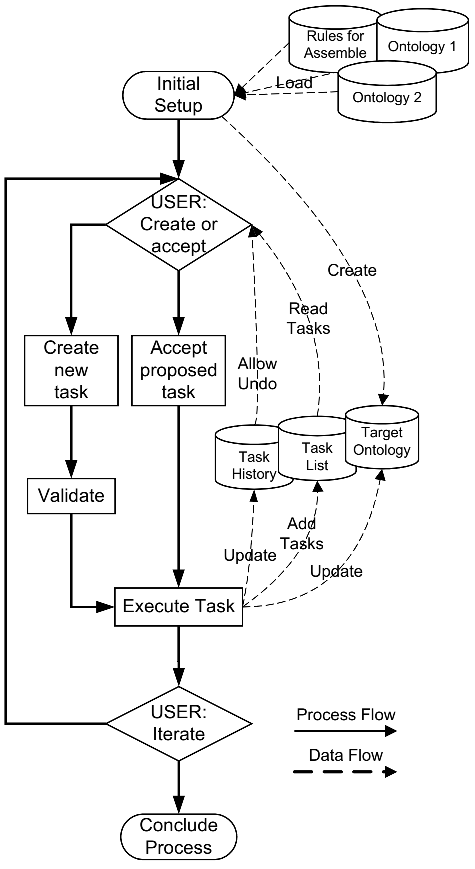

Fig. 4. Assembly main process

By the Execute Task step the corresponding changes are made to the target ontology. Thereafter, the user decides either to pursue another iteration or to go to Conclude Process and accept the current Target Ontology as the final version.

The result of assembling the time-less domain ontology with the temporal theory is the product of automatic proposed tasks and the human interaction regarding the assembling of the three main ontology components, namely, concepts, frame-like relations and general axioms.

III. Proposal

Many projects [12] [6] are developing automatic matching algorithms through distinct ontology dimensions: linguistic [10] [2], clustering [2] [6], graph-based analysis [13], query-based mapping [12].

While the combination of different ontology's dimensions into the matching phase is common, no research work currently exists in encompassing the formal orthogonal elements provided by the assemble process into the matching phase.

The proposal process would operate in the following order:

1. Assemble phase. During this phase the orthogonal ontology is integrated with the domain ontology, which reveal extra semantic elements;
2. Matching phase. During this phase extra semantic elements are elements upon which is possible to reason, infer or better precise semantic relations between two ontologies' entities.
3. Bridging phase. According to the new similarity measures, the services are able to better decide about the validity and characteristics of a semantic relation.

Two types of scenarios are addressed in the envisaged approach. Consider the following concepts for later detailed explanations on both scenarios.

$$\begin{aligned} DO &= \textit{Domain Ontology} && (1)\\ OO &= \textit{Orthogonal Ontology} && (2)\\ MDO1 &= \textit{Medical Domain Ontology 1} && (3)\\ MDO2 &= \textit{Medical Domain Ontology 2} && (4)\\ TOO1 &= \textit{Time Orthogonal Ontology} && (5)\\ TOO2 &= \textit{Time Orthogonal Ontology} && (6) \end{aligned}$$

For instances SNOMED (The Systematized Nomenclature of Medicine)[3] or ICPM (International Classification of Procedures in Medicine)[4] would be some examples for *MDO1* and *MDO2*, while Time-DAML [8] or SUMO [14] would be examples for for *TOO1* and *TOO2*.

A. Common orthogonal ontology

In this type of scenario, the time orthogonal ontology ($TOO1$) is used in the assemble process (7) (8). The resulting ontologies ($A1$ and $A2$) are later mapped (10).

$$\begin{aligned} A1 &= \textit{assemble}(MDO1, TOO1) && (7)\\ A2 &= \textit{assemble}(MDO2, TOO1) && (8)\\ M1 &= \textit{mapping}(MDO1, MDO2) && (9)\\ M1' &= \textit{mapping}(A1, A2) && (10) \end{aligned}$$

From the relations established between domain ontologies and general ontology, it would be possible to infer or precise the mapping relations between domain ontologies, so one can conjecture that $M1'$ (10) would be semantically more complete and accurate than $M1$ (9).

In the example of figure 5 two ontologies from substantially different domains are to be mapped. The transportation/warehousing ontology describes *Place* and *Vehicle* concepts and the real estate ontology specifies *Plant* and *House*. Complementarily, the 'Locations' general ontology defines the *Location* concept, and its two mutually exclusive sub-concepts *Internal Location* and *External Location*.

The semantic relation between *Place* and *House* would be easily perceived through linguistic-based matchers, but the relation between *Place* and *Plant* is difficult to automatically find since only a minimal relation exists between "plant" and "manufacturing plant" and from this to "place".

However, the assemble process would provide extra knowledge, which can be exploited in the mapping process. The assemble process determines that the transportation/warehouse ontology's concept *Place* is a *Location* and that the real estate ontology's *Plant* is an *External Location* and that the *House* concept is a *Internal Location*.

Considering that:

$$\begin{aligned} &\textit{isa}(\textit{Place}, \textit{Location})\\ &\textit{isa}(\textit{House}, \textit{Internal Location})\\ &\textit{isa}(\textit{Plant}, \textit{External Location})\\ &\textit{match}(\textit{Place}, \textit{House}) \end{aligned}$$

the envisaged matcher would suggest the match:

$\textit{match}(\textit{Place}, \textit{Plant})$

Further, if both matches finally give raise to the following semantic bridges:

$$\begin{aligned} sb1 &= \textit{semanticBridge}(\textit{Place}, \textit{House})\\ sb2 &= \textit{semanticBridge}(\textit{Place}, \textit{Plant}) \end{aligned}$$

then the matcher would further suggest the application of the constraint axiom:

$$xor(sb1, sb2)$$

derived from the fact that *Place* is one of *House* or *Plant*, and therefore, no instance of *Place* can be transformed into both *House* and *Plant* instances.

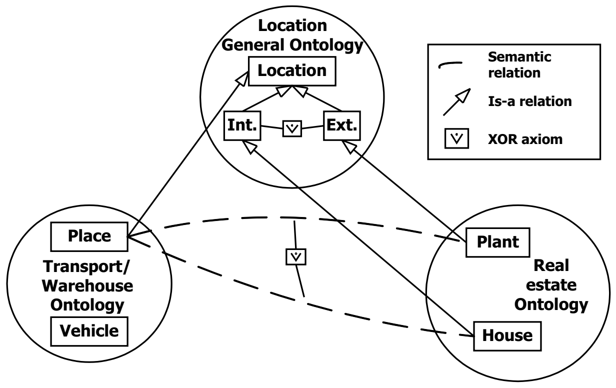

Fig. 5. Common General Ontology used

[3]http://www.snomed.org/
[4]http://icpm.sourceforge.net/

B. Different orthogonal ontology

Let's consider another type of scenario. In addition to the ontologies previously referred (see (3) to (6)) two different orthogonal ontologies about time are mapped together into $M2$ (11). Furthermore both orthogonal ontologies are used independently in two processes of assembling time into two different domain ontologies (see (12) and (13)).

$$\begin{aligned} M2 &= \textit{mapping}(TOO1, TOO2) \quad (11)\\ A3 &= \textit{assemble}(MDO1, TOO1) \quad (12)\\ A4 &= \textit{assemble}(MDO2, TOO2) \quad (13)\\ M3 &= \textit{mapping}(A3, A4) \quad (14)\\ M3' &= \textit{mapping}(A3, A4, M2) \quad (15) \end{aligned}$$

Finally, $M3$ is the output from the mapping between the new domain ontologies assembled with time (14).

Lets consider a short example using the previous ontologies, where both temporal ontologies uses a temporal class for representing a time span, *TOO1:Period* and *TOO2:Interval* with roughly same properties and the mapping output $M2$ states that such classes are semantically equivalent. Such knowledge can be used during the mapping between $A3$ and $A4$ when considering any classes related with *TOO1:Period* and *TOO2:Interval*. So one can conjecture that $M3'$ (15) would be semantically more complete and accurate than $M3$ since it would benefices from extra knowledge contained in $M2$.

C. Goals

From previous descriptions and scenarios, four actuation vectors are envisaged:

1. Human-based systematization of the orthogonal knowledge used in the matching and bridging phases, and derive useful matching and bridging patterns;
2. Combine the derived patterns with both the orthogonal ontologies and the assemble process in order to improve the ontologies and the process;
3. Conceive comparison algorithms and implement the corresponding matchers;
4. Conceive new or expand existent services to exploit the new matchers and bridging patterns.

IV. Related Work

In the past a variety of approaches were proposed for reducing the complexity of engineering a rule-based system, e.g. by task analysis [20], or an ontology-based system, e.g. by developing with patterns [5, 23, 9] or developing subontologies and merging them [15, 17]. As different as these methods are, they may be characterized by subdividing the task of building a large ontology by engineering, re-using and then connecting smaller parts of the overall ontology.

Though FONTE [19] shares its goal with these methodologies is its rather different in its operationalization. FONTE does not aim at a partitioning and re-union (by merge or align with recognition of similarities) of the problem space, but rather by a factorization into primordial concepts and a subsequent combination $\otimes$ that is more akin to a Cartesian product than a union of ontologies.

Despite the difference, one may note that FONTE implements an iterative and interactive approach which was previously successfully adopted in sophisticated tools for merging ontologies [15, 13, 11]. Also, FONTE does not substitute these other methodologies, rather we envision that one wants to separate the target ontology to be built into different (possibly overlapping) domains *as well as* into time-less and temporal subontologies. The two ways of carving up the engineering task need different, complementary methodologies.

There is a rich set of languages and systems that deal with intricate reasoning over time and objects (cf., e.g., temporal description logics [1], temporal databases [3], or event calculus [4]). To our knowledge, however, there has not been a methodology that helped the ontology engineer to build ontologies that included temporal theories.

V. Conclusions and Future Work

In what concerns FONTE we have only studied the assembly of time into a given ontology so far, but we have reason to conjecture, *(i)*, that FONTE may also be applied to integrate other important concepts like space, trust, or user access rights — concepts that pervade a given ontology in intricate ways and necessitate management of engineering complexity; and, *(ii)*, building on Wiederhold's idea of an ontology algebra, the proposed operator $\otimes$ might perhaps be further utilized for having ontologies interoperate in manifold ways.

Regarding the proposal of combining ontologies mapping and assembling processes we can, so far, refer the following advantages:

- Increase matching accuracy and mapping results, once a new comparison dimension is added;
- Increase matching performance by the reduction of the matching and bridging problem space;
- Increase quality of both the orthogonal and assemble ontologies by revisions suggested by the matching and bridging phases.

Acknowledgments

This work was partially supported by FCT (Portuguese Science and Technology Foundation) through programs SANSKI (POCTI-41830) and ONTOMAPPER (POSI-41818).

We are also in indebted to our colleagues in AIFB, particulary to Steffen Staab for his ideas and work about the assemble process and to Gerd Stumme and Julien Tane for their collaboration.

VI. References

[1] Alessandro Artale and Enrico Franconi. A temporal description logic for reasoning about actions and plans. *Journal of Artificial Intelligence Research*, 9:463–506, 1998.

[2] Sonia Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.

[3] Claudio Bettini, Xiaoyang Sean Wang, and Sushil Jajodia. Temporal semantic assumptions and their use in databases. *IEEE Transactions on Knowledge and Data Engineering*, 10(2):277–296, 1998.

[4] L. Chittaro and A. Montanari. Efficient temporal reasoning in the cached event calculus. *Computational Intelligence*, 12:359–382, 1996.

[5] P. Clark, J. Thompson, and B. Porter. Knowledge patterns. In *KR2000*, pages 591–600, 2000.

[6] AnHai Doan, Jayant Madhavan, Pedro Domingos, and Alon Halevy. Learning to map ontologies on the semantic web. Honolulu, Hawaii, USA, May 2002. Honolulu, Hawaii, USA.

[7] N. Guarino and C. Welty. A formal ontology of properties. In *Knowledge Acquisition, Modeling and Management*, pages 97–112, 2000.

[8] J. Hobbs. Towards an ontology for time for the semantic web. In *Proc. Workshop on Annotation Standards for Temporal Information in Natural Language,LREC2002*, Las Palmas, Spain, May 2002.

[9] Chih-Sheng Johnson Hou, N. F. Noy, and M. A. Musen. A template-based approach toward acquisition of logical sentences. In *Procs.Intelligent Information Processing 2002 - World Computer Congress*, Montreal, Canada, 2002.

[10] Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. Rome, Italy, 2001. Rome, Italy.

[11] D.L. McGuinness, R. Fikes, J. Rice, and S. Wilder. An environment for merging and testing large ontologies. In *Procs.KR2000*, 2000.

[12] Renne J. Miller, Laura M. Haas, and Mauricio A. Hernndez. Clio: Schema mapping as query discovery. pages 77–88, Cairo, Egypt, 2000. Cairo, Egypt, Morgan Kaufmann.

[13] P. Mitra and G. Wiederhold. An algebra for semantic interoperability of information sources. In *Proc.2nd.IEEE Symp. on BioInformatics and Bioengineering, BIBE 2001*, pages 174–182, Bethesda, MD/USA, 2001.

[14] I. Niles and A. Pease. Toward a standard upper ontology. In *Procs of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*, 2001.

[15] N. Noy and M. Musen. Prompt: Algorithm and tool for automated ontology merging and alignment. In *Procs.AAAI-2000*, 2000.

[16] A. Preece and R. Shinghal. Foundation and application of knowledge base verification. *International Journal of Intelligent Systems*, 9(8):683–702, 1994.

[17] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.

[18] J. Santos, C. Ramos, Z. Vale, and A. Marques. Verification & validation of power systems control centres kbs. In *Procs.IASTED Artificial Intelligence and Applications (AIA2001)*, pages 324–329, Marbella, Spain, September 2001.

[19] J. Santos and S. Staab. FONTE - Factorizing ONTology Engineering complexity. In *K-Cap'03, The Second International Conference on Knowledge Capture*, pages 146–153, Florida/USA, October 2003.

[20] G. Schreiber, Akkermans H., Anjewierden A., R. Hoog, N. Shadbolt, W. Van de Velde, and B. Wielinga. *Knowledge engineering and management. The CommonKADS Methodology*. MIT Press, 1999.

[21] N. Silva and J. Rocha. Mafra an ontology mapping framework for the semantic web. University of Colorado at Colorado Springs (UCCS), Colorado Springs (CO), USA, June 2003. University of Colorado at Colorado Springs (UCCS), Colorado Springs (CO), USA.

[22] N. Silva and J. Rocha. Semantic web complex ontology mapping. *Web Intelligence and Agent Systems Journal*, (accepted for publication), 2004.

[23] S. Staab, M. Erdmann, and A. Maedche. Engineering ontologies using semantic patterns. In *Procs. IJCAI-01 Workshop on E-Business & the Intelligent Web*, 2001.

[24] S. Staab and A. Maedche. Knowledge portals: Ontologies at work. *AI Magazine*, 22(2):63–75, 2001.

[25] G. Wiederhold. An algebra for ontology composition. In *Proc.Workshop on Formal Methods*, pages 56–61, Monterey, 1994.