Reasoning on Dynamically Built Reasoning Space with Ontology Modules

Fabio Porto
EPFL - Ecole Polytechnique Fédérale de Lausanne
School of Computer and Communication Sciences
Database Laboratory, 1015 Lausanne, Switzerland
Fabio.porto@epfl.ch

Abstract.

Several applications require reasoning over autonomously developed ontologies. Initially conceived to explicit the semantics of a certain domain, these ontologies become a powerful tool for supporting business interactions, once heterogeneities have been solved and inconsistencies eliminated. Unfortunately, a stable coherent logical state is hard to maintain in such an environment, due to normal evolution carried out independently over individual ontologies. As a result, reasoning over autonomously developed ontologies has to face with both heterogeneity and inconsistency, in order to assure correct answering. In this paper we study the problem arising in these settings. We propose an incremental reasoning approach based on a virtual reasoning space that is filled with relevant ontology entities as query answering progress. We show how to identify the set of relevant entities with respect to a user query using a set theory approach and illustrate the solution with a use case exploring the web service discovery scenario.

1. Introduction

The use of ontologies to formally describe a domain has been adopted by applications in various areas like bioinformatics[1,12,13], business [14,15], transportation [16,17] etc.. Such increasing interest on ontologies to support all kinds of web related and web agnostic applications do not, however, point to a future global and uniform ontology [18] but rather to a set of autonomously specified ones. In spite of considering this characteristic as precluding the use of such ontologies, many authors have investigated a more general strategy for reasoning over ontologies that are distributed and autonomously specified [2,7,19,20,21].

Reasoning over distributed and autonomous developed ontologies has to face a number of new challenges. First, current reasoners [23] consider ontology as forming a single logical theory. Unfortunately, both distribution and autonomy adversely contribute to such a view. Therefore in order to use current reasoning software the set of autonomous developed ontologies must be aligned and integrated into a

single consistent ontology. Second, as in the context of database integration [22], and to allow building a single logical theory, definition on different ontologies must be aligned by the use of correspondence expressions. Thirdly, the set of involved ontologies may get to a quite voluminous amount of data. As a result, a naïve solution of transferring all ontologies to a location and then proceed with local reasoning does not scale up. Finally, autonomously defined ontologies may assert contradictory definitions, which some authors classify as conflicts in the integration process. Conflicts identification is, in fact, a tool for fixing correspondence assertions and applying ontology alignment. So, reasoning under this setting should be capable of identifying such conflicts and acting appropriately.

In this paper, we propose a new strategy for building a single ontology out of autonomously developed ones to answer an ontology query. We consider an ontology together with a set of correspondences with other ontologies forming a ontology module. Ontology queries submitted to modules are answered by reasoning over a dynamically built reasoning space comprising relevant ontology entities captured among autonomous developed ontologies. We give some initial ideas on how to dynamically build a reasoning space and point to further research issues.

The rest of this paper is structured as follows. Section 2 presents the concepts of ontology spaces and ontology modules. Next, section 3 develops the strategy of building a reasoning space to answer reasoning queries over an ontology space. Section 4 uses a scenario of web services discovery to illustrate the approach. In section 5, we comment on relevant related work and, finally, section 6 gives our conclusions and point to some future work.

2. Ontology Space and Modules

Autonomously developed ontologies emerge quite naturally in different business areas. However as business evolves, interactions among partners promote the extension of each one's activities towards a network of interrelated process and data. If automation is required to support the business process, the independent developed ontologies may prove useful in solving semantic misunderstandings by offering a wider semantic cover for reasoning tasks.

We name a set of autonomously specified ontologies over which an hypothetical reasoner could evaluate an ontology query an *ontology space* (OS). Giving two ontologies taking part in a *OS*, we say that they intersect if there is a known correspondence assertion associating entities in both ontologies.

The set of entities specified in a ontology together with a set of correspondences expressed with entities in other ontologies define an ontology module (M). The underlying ontology of a module is named its *base ontology*. An ontology entity in a module is either defined in its *base ontology*, local entity, or added to it by an equivalence correspondence with an external entity, specified in a different ontology. The concept of modules is similar to context in C-OWL [2].

> **Definition 1**: A *module* is a set Mo = <id, D, L, C, O_b, Os>, where *id* corresponds to a Unique Resource Identifier (URI) for the module, *D* is the description of the module, either expressed in natural language or by means of an ontology language; *L* is the ontology language used in *Mo*; *C* is a set of correspondences (defined below) associating local entities with entities defined in external modules; O_b is the base ontology and *Os* is the set of external ontologies to which correspondences with local entities are specified.

The ontology description should aid both humans and machines in selecting modules. Such descriptions may include domain characteristics, non-functional properties, and assumptions. The latter can be used, for instance, in deciding which modules to consider in answering a query.

Definition 2 below specifies valid correspondences between ontology entities[2].

> **Definition 2:** An ontology correspondence is a relation in one of the following forms:
>
> - $C \equiv D$ (for class equivalence)
> - $C \subseteq D$ (for subsumption)
> - $C \supseteq D$ (for superset)
> - $R \equiv S$ (for relationship equivalence)
> - $v \equiv t$ (for instance equivalence)

where (C, R, v) and (D, S, t) are, respectively, local and external entities with respect to a module. *C* is of type class, *D* is a class expression of the form $f(t_1,...,t_n)$, where the terms t_i are either class names or class expressions and *f* is an n-ary class builder operator, *R* and *S* are ontology relationships, and *v* and *t* are instances [2].

Correspondences are specified from a module designer point of view. They contribute to the semantic autonomy of each module by giving local interpretation to external entities, with no impact on their semantics in the original ontologies. We further consider that the ontology correspondences complements the base ontology's definitions and can be locally validated indicating eventual conflicts.

We also define a *peer* P=<Mo,QL> that models a software component capable of answering ontology queries expressed in *QL* language over an ontology module Mo. A *peer system* is a set PS= $\cup$ P_i.

3. Reasoning Space

We use the term *reasoning space* (RS) to denote a virtual ontology that is dynamically built to answer an ontology query over an ontology space.

A reasoning space includes the base ontology associated to a module that receives the query and complementary elements gathered from external ontologies. Entities of a reasoning space share the same ontology language and form a single ontology.

Definition 3: A *Reasoning space* RS is defined as: RS $\subseteq$ {O $\cup$ C}, where O is an ontology space and C is the set of correspondences associating elements in O.

Definition 4: We also define a reasoning space *mapping function f(Q,RS,O):RS'* that given: a ontology query *Q*, a reasoning space *RS* and a ontology space *O*, produces a new *reasoning space RS'*.

The mapping function *f* expands *RS* during query evaluation. Reasoning on a RS is done incrementally as relevant entities in external ontologies are identified and added to it. As soon as the query is decided, the incremental process ceases.

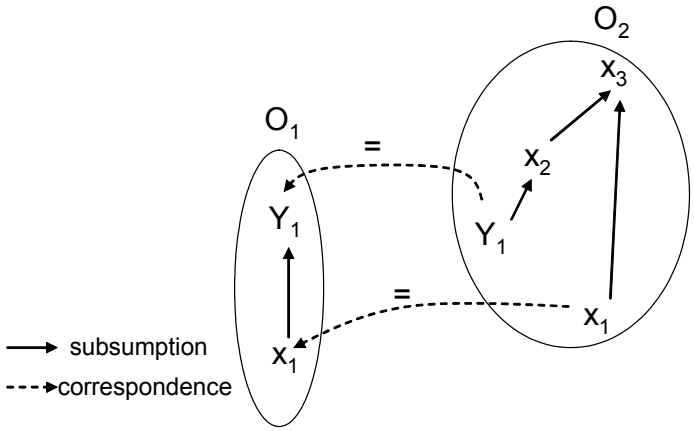

Figure 1 Ontology space

Let us motivate the discussion on *reasoning space* by aid of a simple example, as illustrated in Figure 1. The picture presents an ontology space O, comprised of two ontologies, O_1 and O_2. Module $M_1=\{O_1,C_1\}$ includes its base ontology O_1 and a set of equivalence type of correspondences C_1, associating entities defined in O_2. One may clearly identify that the complete logical theory is inconsistent as the subsumption relation between O_1:y_1 and O_1:x_1 should also hold in O_2 as a result of C_1. Unfortunately, as a result of the evolution of autonomously managed ontologies, we should expect that inconsistencies like this one are prone to emerge and should be considered when reasoning over the ontology space.

In order to complete the example, a ontology query *Q*, $Q= x_1 \subseteq x_2$, is submitted[1] to ontology module M_1. Query *Q* can not be decided using uniquely entities specified in M_1 base ontology O_1, therefore the mapping function $f(Q,O_1 \cup C_1,O):RS_1$ is computed to extend the original reasoning space comprised initially of the union of ontology O_1 and the correspondence set C_1. The mapping function *f* identifies a set of relevant entities in O_2 to be included into the *reasoning space* of query *Q*. Relevant entities are those in O_2 associated to entities in C_1 that appear in *Q*, *RE*=$\{ y_1, x_2, x_3, y_1 \subseteq x_2, x_2 \subseteq x_3, x_1 \subseteq x_3 \}$. The *reasoning space RS* is augmented with relevant entities in R, $RS` = O_1 \cup R$, and

[1] We consider the existence of a query answering system on top of each module forming a P2P network

reasoning over RS' can proceed. Having all the entities relevant for query *Q*, RS' has sufficient knowledge for deciding the query. As a matter of fact, *RS'* will bring up the existing inconsistency in the ontology space, providing an opportunity for alignment between ontologies and correspondences[2].

3.1. Ontology Query Model

We consider boolean DL conjunctive queries where users want to check on satisfiability with respect to a ontology space. These query types are important for applications like web service discovery, where a matching process requires to verify subsumption and equivalence between goals and web service description terms, as well as satisfiability of *instance of concept* expression [4].

Our approach is based on set theory, as adopted in [25], in the context of web service discovery. In this context, a query expresses a conjunction of disjoint sets of objects.

Definition 4: An ontology query is in reduced clause form RCF [25] if given $Q= q_1 \wedge q_2 \wedge \ldots \wedge q_n$, where q_i is a clause modeling a set of objects, then $q_i \cap q_j = \varnothing$, $i \neq j$, $1 \leq i,j \leq n$.

In our example, the query $Q=x_1 \subseteq x_2$ includes a single clause, restricting the concept x_2.

A query in RCF is satisfied if we can prove that each of its disjoint sets is a subset of some set of objects in RS.

3.2. Finding Relevant Entities on the Ontology Space

As discussed above, the mapping function identifies relevant entities on the ontology space to be considered in extending the *reasoning space*. A strategy for identifying the set of relevant entities is the objective of this section.

Identifying relevant entities is achieved in two steps. In the first step, we check for relevant correspondences in the current reasoning space

[2] We do not address in this paper solutions to conflicting situations.

and, in the second step, a new query for obtaining relevant entities is submitted to the respective ontology module.

Definition 5: a relevant correspondence defines a set of objects with a non empty intersection with a RCF query clause.

As an example, for query *Q* and correspondences $C_1=\{O_1: x_1 \equiv O_2: x_1, O_1: y_1 \equiv O_2: y_1\}$, we have that $x_1 \subseteq x_1$ and $x_1 \subseteq y_1$. Therefore the relevant correspondence set $RC=C_1$.

Next, we need to query the corresponding ontology modules for relevant entities. Similarly with Definition 5, the set of relevant entities in ontology modules, RE, are those concepts and roles whose corresponding object set intersects with objects in the RC set.

In our initial example, $RE= O_2 \cap RC$, thus $RE= \{y_1, x_2, x_3, y_1 \subseteq x_2, x_2 \subseteq x_3, x_1 \subseteq x_3\}$.

3.3. Answering Queries over the Reasoning Space

A reasoning space is obtained by successively extending a prior version. The extension includes the relevant entities obtained in the process as described in section 3.2 and the correspondences fetched from the target module.

Once obtained, a traditional reasoner evaluates the query over the reasoning space. The process finishes when, either the query has been decided or there is no more possible extension of the reasoning space.

In case an inconsistency is detected a user intervention may be requested to allow for process continuation.

3.4. Dealing with Global Interpretation

In a ontology space made of autonomous independent ontologies, reasoning has to consider how to interpret definitions to which explicit correspondences have not been specified. In the running example, analyzing the satisfiability of query $Q=x_1 \subseteq x_2$, depends on the given interpretation for both x_1 and x_2. If a local interpretation is assumed, by

prefixing each ontology entity with a local identification, then satisfiability is only achieved if explicit correspondences associate query terms interpretation with ontology entities used for reasoning.

On the other hand, one may be interested in possible answers for the reasoning query. In this scenario, entities computed as relevant that present the same term are considered as having an implicitly equivalence correspondence. The motivation for such assumption is that relevant entities are taken from the intersection set of query clause with relevant correspondences (see section 3.2) in the remote ontology. This reinforces that both terms share the same semantic context and, thus, may be equivalent. Producing possible answers may include providing users with a list of assumed correspondences, so that further processing may analyze its pertinence.

4. Applying the Reasoning Space Approach into a Use Case

In this section, we illustrate the procedure for reasoning over a *reasoning space* as presented in section 3 above. We consider a use case in which users search for Web services that provide car rental services.

We take the approach presented in [3] in which Web service functionality (or capability in WSMO terms [4]) is described by means of *conjunctive formulae* [5] indicating the objects involved in the functionality provided by the Web service and relationship between these objects. Correspondingly, user queries are grounded conjunctive queries that express the desired service, which in WSMO is called a *Goal.*

Thus, finding a Web service that satisfies the user corresponds to matching the user goal against descriptions of Web service functionality. Unfortunately, very often, terminologies used in describing the goal and the web service functionality may be different. This is where the ontology space comes into the game. It provides the means to verify the correspondences between terms used in the goal and web service functionality definitions.

In this context, let us consider an ontology space $OS=\{O_1,O_2\}$, with its corresponding modules $M=\{M_1, M_2\}$ that are used by the matching algorithms to eliminate ambiguities and heterogeneities in between goal and web service description terminology.

An agent looking for booking a *sportscar* in the city of *Lausanne*, as part of a tourism package, would initiate a Web service discovery process by submitting a corresponding goal to the system. Let's assume that a single Web service has been advertised by offering as one of its functionalities the rental of a set of car models in Europe.

The agent's goal g and Web service description ws would be expressed as below:

g = carRental	and	model(sportscars)	and	place(Lausanne)
ws = carRental	and	model(Ferrari)	and	place(Europe)

Based on this input, the discovery process initiates a matching function which analyzes the correspondences between predicates *carRental*, *model* and *place* in *g* and *ws*. These, however, cannot be directly matched because of the semantic heterogeneity between the goal and the web service description. Ontological support is needed to overcome the semantic gap. Thus, the matching function submits a query to module M_1 to find out whether *Ferrari* is a model of *sportscar* and *Lausanne* is a place in *Europe*, which would lead to a successful match between the goal *g* and the web service description *ws*. The query to M_1 is expressed as:

q: Ferrari $\subseteq$ sportscar and Lausanne $\subseteq$ Europe

The reasoning task is evaluated considering the module M1=<1,d,l,C_1,O_1,O_2>, exemplified in Tables:T1 and T2.

Table 1. Ontologies O1 and O2

O_1	O_2
Concept(Car)	Concept(vehicle)
Concept(turbo_engine_car)	Concept(sportscar)
Concept(Lausanne)	Concept(Ferrari)
Concept(EU)	Concept(Europe)
Turbo_engine_car $\subseteq$ Car	sportscar $\subseteq$ vehicle
Lausanne $\subseteq$ EU	Ferrari $\subseteq$ sportscar

Table 2. Ontology Correspondence Definitions c_1

c_{11}: O_1: turbo_engine_car $\supseteq$ O_2: Ferrari
c_{12}: O_1: EU $\equiv$ O_2: Europe

Query *q* is in RCF, presenting clauses t_1= Ferrari $\subseteq$ sportscar and t_2= Lausanne $\subseteq$ Europe. The evaluation of *q* initially considers the *reasoning space* RS= $O_1 \cup C_1$. In this context, clause t_2 can be decided by using correspondence c_{12}, the same not being observed with respect to the clause t_1 that remains undecided. The evaluation of *q* proceeds by extending the initial reasoning space towards relevant entities defined in O_2, with respect to t_1.

The logical expression in t_1 specifies the set of objects where Ferrari is a subset of sportscar. Analyzing the set of relevant correspondences in C_1, c_{11} is identified as providing the set of objects where Ferrari is a *turbo_engine_car*, thus $c_{11} \cap t_1 \neq \varnothing$ and is chosen to compose the set of relevant correspondences. The *relevant entities* of O_2 with respect to c_{11} is obtained by evaluating RE= $\{O_2 \cap c_{11}\}$ => {Concept(vehicle), Concept(sportscar), Concept(Ferrari), Ferrari $\subseteq$ sportscar, sportscar $\subseteq$ vehicle}.

Finally, the reasoning space RS is augmented with $RS = RS \cup$ RE and the evaluation of t_1 can take place.

An attentive reader may argue that the query rewriting approach [6] could be used to decide on query *q* without the burden of formulating a global *RS*. This would be the case if we could guarantee consistency over ontologies in the ontology space. As discussed in section 1, conflicting definitions among participating ontologies may raise as a result of autonomous ontology evolution. In this context, if queries are rewritten and evaluated over single ontologies, such conflicts would be impossible to detect, bringing eventually to users contradictory answers, which justifies the proposed approach for reasoning over a single logical theory that is incrementally extended.

5. Related Work

Ontology modularization is a new research issue that has attracted the attention of researchers dealing with large and distributed ontologies. In

this section, we summarize some of the works that have been developed in this context.

The composition approach has been targeted by C-OWL [2], which aims to support the scale-up of large and distributed ontologies by specifying an ontology as the result of linking autonomously developed ontologies. In C-OWL, a set of independent ontologies form a context OWL space, where each ontology O_i is enriched by components defined on external ontologies O_j mapped according to O_i interpretation. A bridging language is specified for defining mappings (coordination) with some predefined associations like: subsumption (c-owl:into), equivalence(c-owl:equivalence), containement(c-owl:onto), disjunction (c-owl:incompatible) and intersection (c-owl:compatible). A consequence of the C-OWL approach is that by defining their own mapping rules, a local ontology may get inconsistent but would not affect the consistency of the remaining ontologies. We adopt the same principles presented in C-OWL regarding correspondences among ontologies.

Another interesting approach for reasoning over autonomously developed ontologies is proposed in the context of the *WonderWeb* project [7].The requirements suggested in the work comprise: loose coupling, self-containment, and integrity. The first point, loose coupling, corresponds to the idea of autonomously developed ontologies, regarding language specification and instance interpretation. The second issue regards autonomous reasoning. In this sense, a module should offer a complete reasoning context for a certain application. Finally, integrity is associated to a correct reasoning in the presence of autonomous modules. Based on these principles, the work proposes a modularization approach where self-contained modules are cross connected through materialized views expressed as conjunctive queries. The connection of modules thereof is obtained by defining an equivalence relation between a concept in a module (local ontology) and the result of evaluating a query on an external ontology. The result produced by evaluating the connection view is materialized into the local ontology as new axioms, contributing to the definition of a self-contained module. A procedure for managing updates in an external ontology definition is also proposed. The authors argue that the proposed mapping language, expressed as conjunctive queries, is more expressive than standard methods of directing referencing objects in an external ontology, such as adopted in OWL import strategy [8].

The problem of semi-automatically defining modules from an initial single ontology is being studied by Menken, Stuckenschmidt and Wache [9] within the context of KnowledgeWeb. In their work, a graph based approach for representing relevant ontology definitions provides for some algorithmic and heuristic analysis leading to suggestions on the partition of a ontology into a set of modules. The approach targets to achieve modularization by identifying clusters of semantically related concepts. S*emantic relatedness* is extracted by representing in the dependency graph concepts as nodes and their relationship with other concepts and properties as weighted edges. In general, the approach is structured into the following steps: (1) build a dependency graph; (2) determine the strength of dependencies; (3) identify modules and (4) improving partitioning. A tool has been put into place that integrates and extends different components. It captures ontologies stored in a Sesame repository and builds the dependency graph. Next, an external program computes the clusters based on the dependency graph and some user defined parameters. Further cluster analysis uses a network analysis tool to compare different partitioning propositions.

In [21] a proposal for reasoning on distributed ontolgies with correspondences is presented. The goal is to define a theoretical solution for the problem of global subsumption and to propose a P2P implementation that assesses the practical adequacy of the proposal. Their main result is to prove that sub-sumption between remote ontology entities can be proved using local sub-sumption relationships and correspondences between relevant entities on both ontologies. This leads to global entailment and offers a solution for the problems we investigate in this paper. The strategy can be seen as based on query rewriting approach, similar to what is done in database integration, with distributed reasoning applied using distributed local tableau. Local inconsistencies are treated as holes[2].

6. Conclusion

Reasoning over distributed and heterogeneous ontologies is not an easy task. First, there are no currently available distributed reasoners. Second, keeping correspondences between ontology entities up to date is hard as ontologies evolve. Third, as ontologies cover more complex domains their size augments precluding a complete transfer of whole

ontologies to the queried peer. Finally, inconsistencies among ontologies may offer users contradictory answer that would be hard to detect once the whole result has been produced.

In this paper, we presented a strategy for reasoning over a set of autonomously managed ontologies with correspondences defining local interpretations for foreign defined ontology entities. In our approach, a reasoning space is built including relevant ontology entities, with respect to a ontology query, found in foreign ontologies. Relevant entities are obtained by computing intersections among ontology entities and query clauses. Entities thus after discovered fill the reasoning space allowing the use of efficient and available reasoner tools.

The approach presents solutions to all identified problems but also brings to light new questions. As a matter of fact, deciding on inconsistencies on such a autonomous settings is not easy as it has been discussed with respect to non explicit correspondences. Clearly, a more precise comparison of our approach with other distributed ontology reasoning based on query rewriting [21] is of primordial importance to evaluate the benefits of building a reasoning space. This is in our list of future work. We also plan to implement our approach in a P2P system developed in the context of the DIP project. Finally, we also want to investigate a cost model for expanding the reasoning space. The main intuition is that there are innumerous equivalent paths to follow in exploring the ontology space. A cost model based on previous reasoning tasks and statistics regarding individual ontology entities should certainly contribute to reduce the query elapsed-time.

References

1 M. Ashburner,C. Ball, J.A. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T. Eppig, M.A. Harris, D.P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G.M. Rubin, G. Sherlock, Gene Ontology: tool for the unification of biology, Nature Genetics, V(25), 2000.

2 Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini and Heiner Stuckenschmidt, C-OWL: Contextualizing

Ontologies, 2nd Intl. Semantic Web Conference, Sanibel Island, Florida, USA, pp. 164—179, 2003.
3 U. Keller, R. Lara, A. Pollares, I. Toma, M. Kifer and D. Fensel, ”WSMO Web Service Discovery”, D5.1 v 0.1, WSML Working Draft, November, 2004.
4 H. Lausen, D. Roman, and U. Keller, Web Service Modeling Ontology – standard (WSMO-standard), v. 1.0, working draft, DERI, 2004. http://www.wsmo.org/2004/d2/v1.0.
5 S. Ceri, G. Gottlob, L.Tanca, Logic Programming and Databases, Springer-Verlag, 1990.
6 D. Calvanese, G. De Giacomo, M. Lenzerini, and Moshe Y. Vardi, “View-based Query Processing: On the Relationship between Rewriting, Answering and Lossleness, ICDT, LNCS 3363, pp. 319-334, 2005.
7 Heiner Stuckenschmidt, Michel Klein, Modularization of Ontologies, WonderWeb: Ontology Infrastructure for the semantic Web, Del 21, V.0.6, May 14,2003 .
8 Sean Bechhofer, Frank van Harmelen, Jim Hendlera, Ian Horrocks and et all, OWL Web Ontology Language Reference, Recommendation 10 Feb. 2004, W3C Recomendation,http://www.w3.org/TR/owl-ref/, last access, April 2005.
9 M. Menken, H. Stuckenschmidt, H. Wache, private communication, 2005.
10 Christelle Vangenot, Christine Parents and Stefano Spaccapietra, Modeling and manipulating multiple representations of spatial data, Proceedings of the International Symposium on Spatial Data Handling (SDH 2002), July 9-12, Ottawa, Canada, 2002.
11Mustafa Jarrar, Ontology Modularization and composition,Workshop on Ontology Modularization and context, Dec. 14, Brussel, Belgium,2004.
12 Sidhu, A. S., T. S. Dillon, et al. (2004). Protein Knowledge Base: Making of Protein Ontology. HUPO 3rd World Congress 2004, Beijing, China.
13 R. Stevens, C.A. Goble, and S. Bechhofer. Ontology-based Knowledge Representation for Bioinformatics. Briefings in Bioinformatics, 1(4):398-416, November 2000.
14 Gruninger, M., Atefi, K., and Fox, M.S., (2000), "Ontologies to Support Process Integration in Enterprise Engineering",

Computational and Mathematical Organization Theory, Vol. 6, No. 4, pp. 381-394.

15 Fox, M.S., Barbuceanu, M., Gruninger, M., and Lin, J., (1998), "An Organisation Ontology for Enterprise Modeling", In Simulating Organizations: Computational Models of Institutions and Groups, M. Prietula, K. Carley & L. Gasser (Eds), Menlo Park CA: AAAI/MIT Press, pp. 131-152

16 http://www.daml.org/ontologies/409, last access 24/05/2005.

17 M. Becker and S. Smith, An Ontology for Multi-Modal Transportation Planning and Scheduling, tech. report CMU-RI-TR-98-15, Robotics Institute, Carnegie Mellon University, November, 1997.

18 Ian Niles, Adam Pease, Towards a Standard Upper Ontology, Proceedings of the international conference on Formal Ontology in Information Systems - Volume 2001, Ogunquit, Maine, USA pp:2-9, 2001

19 M.Bonifacio, P. Bouquet, G. Mameli and M. Nori, "Kex: a Peer-to-Peer solution for Distributed knowledge Management", AAMAS'2003.

20 C. Ghidini and L. Serafini. Distributed First Order Logics. In Proceedings of the Frontiers of Combining Systems,pages 121-139,2000.

21 L.Serafini and A.Tamilin. Distributed reasoning services for multiple ontologies. Technical Report DIT-04-029, University of Trento, 2004.

22 Thomas Devogele, C. Parent, S. Spaccapietra, On spatial database integration, Int. J. Geographical Information Science, v(12), n(4), pp. 335-352, 1998.

23 Volker Haarslev† and Ralf M Oller‡, Racer: An OWL Reasoning Agent for the Semantic Web, In Proceedings of the International Workshop on Applications, Products and Services of Web-based Support Systems, in conjunction with the 2003 IEEE/WIC International Conference on Web Intelligence,Ê Halifax, Canada, October 13, pages 91–95, 2003.

24 K. Aberer, A. Datta, M. Hauswirth, R. Schmidt,"Indexing data-oriented overlay networks", VLDB, 2005

25 B. Benatallah, M. Hacid Al. Leger, C. Rey and F. Toumani, "On automating Web Service Description", VLDB Journal, V(14), pp. 84-96, 2005.