

# CATO - A Lightweight Ontology Alignment Tool

Karin Koogan Breitman, Carolina Howard Felicissimo, Marco Antonio Casanova

PUC-RIO – Pontifical Catholic University of Rio de Janeiro, Department of Informatics,  
Rua Marquês de São Vicente 225, Rio de Janeiro, CEP 22453-900, RJ, Brasil  
{karin, cfelicissimo, casanova}@inf.puc-rio.br

**Abstract.** Ontologies are becoming increasingly common in the World Wide Web as the building block for a future *Semantic Web*. In this Web, ontologies will be responsible for making the semantics of pages and applications explicit, thus allowing electronic agents to process and integrate resources automatically. With the widespread, distributed use of ontologies, different parties will inevitably develop ontologies with overlapping content. The ability to integrate different ontologies meaningfully is thus critical to assure coordinated action in multi agent systems. In this paper, we propose a strategy and tool that allow for totally automatic ontology alignment for the Semantic Web. We subscribe to the view that software agents must act autonomously and should not count on user intervention of any kind. Therefore, our approach to ontology alignment, as opposed to related work that uses semi-automatic or even manual techniques, is to provide CATO, a fast, reliable, user independent tool that allows for autonomous on-line ontology alignment. CATO is publicly available in the Internet and works for any pair of OWL compliant ontologies.

## 1 Introduction

Ontologies are rapidly becoming the *lingua franca* to express the semantics of information on the Web. As envisioned by Tim Berner's Lee [1], in the future, rather than sharing a few domain ontologies, crafted by knowledge engineers, e.g. WordNet [2] and CYC [3], every Web site and application in the Web will have its own ontology. There will be a "*great number of small ontological components consisting largely of pointers to each other*" [4]. Tracing a parallel to the history of the Web itself, Hendler assumes that ontology creation will be done in the same, nearly anarchic, and decentralized fashion as the nearing eight billion Web pages have been created. The result will be a great variety of lightweight ontologies, both built and maintained by independent parties.

His predictions seem to be true, as the number of tools for ontology edition, visualization and verification grow. The best examples are the OilEd and Protégé tools, which sprung from large cooperation projects involving many universities and different countries [5, 6]. With increasing numbers of available books, on line tutorials and materials [7, 8, 9, 10, 11, 12], crafting an ontology today is possibly no harder than creating a Web page ten years ago. In the beginning days of the Internet, Web sites were created exclusively by Web engineers, but as HTML editors progressed, Web page creation became available to most Internet users. Our experience with ontologies

has demonstrated that ontology development is no particularly challenging compared to building other conceptual models used in our software engineering practice, such as class diagrams [13] and  $i^*$  [14]. Evidently, the quality of the resulting model depends on the ability of the person doing the modeling, true to most conceptual models [15]. Indeed, the number of "lightweight" ontologies, i.e., developed by independent groups and organizations rather than by knowledge engineers, is rapidly growing as it can be verified with a visit to some public ontology repository like the DAML's repository at <http://www.daml.org/>.

The co-existence of a multitude of ontologies poses a further problem: semantic interoperability. Open system applications, if anchored in different ontologies, will have to undergo a negotiation process. This operation, referred to as ontology integration, aims at finding an intermediate representation that can be shared by both applications to ensure communication.

In this paper, we focus on the ontology integration problem from a multi agent system perspective. The main contribution of the proposed strategy is to combine well known algorithmic solutions, such as natural language processing and tree comparison [16, 17], to the ontology integration problem. Our universe of discourse is composed of software agents that act autonomously in an open ended environment, driven by their own goals [18]. In order to fulfill their tasks, collaboration with other agents is often required. Different software agents are likely to provide separate ontologies, therefore the ability to align the ontologies into a single representation, that can be shared by different applications, is paramount to ensure communication [19]. To secure autonomous behavior, decisions taken by software agents must be done independently of human intervention. Therefore ontology integration, when necessary, must be done automatically if done at all.

Despite the existence of some strategies and supporting tools for ontology integration, most available techniques are either completely manual or semi-automatic, but all depend on user intervention to some degree. In the next section, we survey ontology integration techniques. In section 3, we introduce the case study that is going to be used to illustrate our approach. In order to allow software agents to negotiate between ontology based applications it is necessary to provide a mechanism that allows a fully automatic integration process. Then, continuing in section 3, we introduce CATO, a tool that implements our ontology alignment approach. Examples of CATO in action are derived from the case study. The listings in this paper were all generated by CATO. In section 4, we discuss the limitations of our strategy. Of course, to guarantee speed and discard user intervention, some commitments had to make. To guarantee a fast response, we limited our approach to lexical and structural comparisons. Much richer analysis could be performed if additional information was used, e.g. restrictions (slots) as it is done in both the Chimaera and Prompt approaches. Furthermore, our alignment strategy is very conservative, in that it discards doubtful matches to preserve reliability. Our conclusions are presented in section 5.

## 2 Related Work

Semantic interoperability among ontologies has been in the research agenda of knowledge engineers for a while now. A few approaches to help deal with the ontology integration problem have been proposed. The most prominent ones are: merging [20], alignment [20, 21], mapping [21] and integration<sup>1</sup> [22]. Merging ontologies results in a unique model containing the sum of concepts from the original ontologies, without indication of its provenance. Aligning ontologies is the identification of the links that hold between concepts from two different inputs. Those links provide the shared semantics of both representations. Alignment is usually done in pairs. The result of the alignment of two input ontologies can be presented either in the format of an intermediate representation (third "aligned" ontology) or as an addition to the markup of the original ontologies. Mapping between two ontologies results in a formal representation that contains expressions that link concepts from one ontology to the second. Finally, ontology integration [22] results in a unique ontology created by assemblage, extension, specialization or adaptation of ontologies from different subject areas. When integrating ontologies it is possible to identify the relationships to the original ontology.

While our work focuses on fully automated merging of ontologies, there are several semi automatic ontology merging tools available. The GLUE system [23] makes use of multiple learning strategies to help find mappings between two ontologies. Given any two ontologies and their instances, the system is able to find nodes that are similar, given a pre defined similarity measurement. It is an automated tool that feeds its result to a semantic interoperability tool that can interpret and make use of its results. Iprompt provides guidance to the ontology merge process by describing the sequence of steps and helping identify possible inconsistencies and potential problems. AnchorPROMPT [21], an ontology alignment tool, automatically identifies semantically similar terms. It uses a set of anchors (pairs of terms) as input and treats the ontology as a directed graph. In this graph, the nodes are the ontology classes and the links its properties. It makes use of similarity measurements and equivalence groups to help detect similar terms. The Chimaera environment [36] provides a tool that merges ontologies based on their structural relationships. Instead of investigating terms that are directly related to one another, Chimaera uses the super and subclass relationships that hold in concept hierarchy to find possible matches. Their implementation is based in Ontolingua editor [24].

---

<sup>1</sup> Please note that we use the term ontology integration as an abstraction that encapsulates all different treatments, including Pinto et al ontology integration approach.

## 3 Ontology alignment with CATO

### 3.1 Overview

In this section, we outline the ontology alignment strategy that CATO implements. CATO takes as input any two ontologies written in W3C recommended standard OWL. An online version of CATO is publicly available at the following address: <http://cato.les.inf.puc-rio.br/>. It was fully implemented in JAVA and uses a specific API (Application Programming Interface) that deals with ontologies, JENA [25]. The listings in this paper were all generated by CATO.

The philosophy underlying our strategy is purely syntactical. We perform both lexical and structural comparisons in order to determine if concepts in different ontologies should be considered semantically compatible. We use a refinement approach, broken into three successive steps.

We first compare concepts lexically in order to identify those that share lexical equivalence. The concepts from both ontologies first go through a lexical normalization process, in which they are transformed to a canonical format. The concepts are then compared, with the aid of a dictionary, in order to find lexically equivalent pairs.

Our assumption is that the use of lexically equivalent terms implies the same semantics, if the ontologies in question are in the same domain of discourse. For pairs of ontologies in different domains, lexical equivalence does not provide guarantee that the concepts share the same meaning.

To solve this problem, our strategy proposes to use structural comparison. Concepts that were once identified as lexically equivalent are now structurally investigated. Making use of the intrinsic structure of ontologies, a hierarchy of concepts connected by subsumption relationships [7], we now isolate and compare concept sub-trees. Investigation on the ancestors (super-concepts) and descendants (sub-concepts) will provide the necessary additional information needed to verify whether the pair of lexically equivalent concepts can actually be assumed to be semantically compatible.

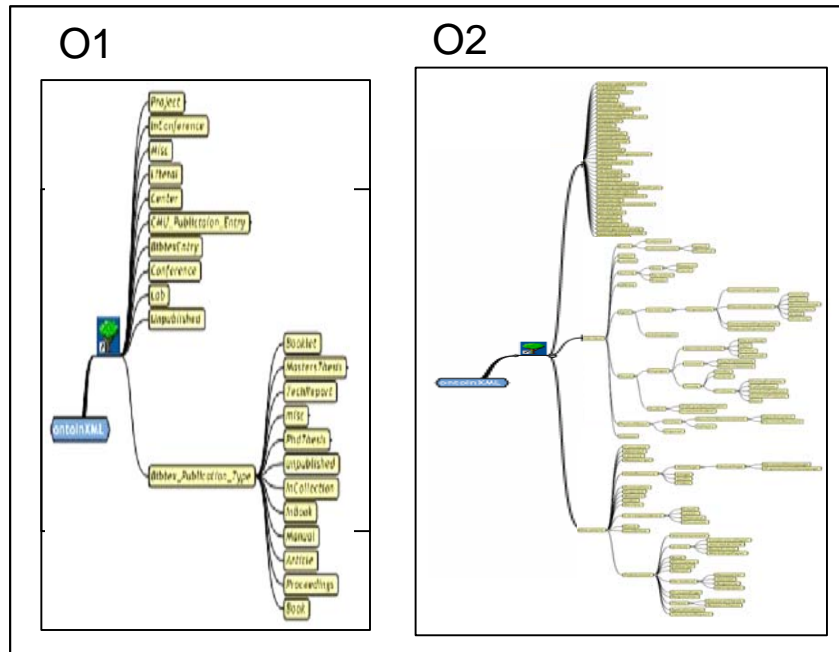
Lexical comparison is done during the first and again during the second step of the strategy. Structural analysis is done in the last two steps of the strategy, detailed in what follows.

The result is an *OWL* document containing *equivalent class* statements (`<owl:equivalentClass>`) that relate the equivalent concepts from the two input ontologies.

### 3.2 Case Study

To illustrate our ontology alignment strategy and tool, we chose two independent ontologies, created by reliable third parties. Both ontologies model the academics domain. The first ontology, CMU RI Publications [26], was developed by the group responsible for the Agent Transaction Language for Advertising Services (ATLAS)

[27] project at Carnegie Mellon University. The second ontology, General University Ontology, was developed by the French company Mondeca SA [28] that belongs to the W3C Web-Ontology's work group. The two chosen ontologies are different both structurally and in the total number of concepts. Figure 1 provides an overall view of the structure of both ontologies. The first ontology is comprised of 25 concepts whereas the second one is comprised of 225 concepts.



**Fig. 1.** The compared ontologies' hierarchy of the case study presented

### 3.3 Detailed description of the proposed strategy

CATO is based in the three step alignment strategy, illustrated in Figure 2 and detailed in what follows.

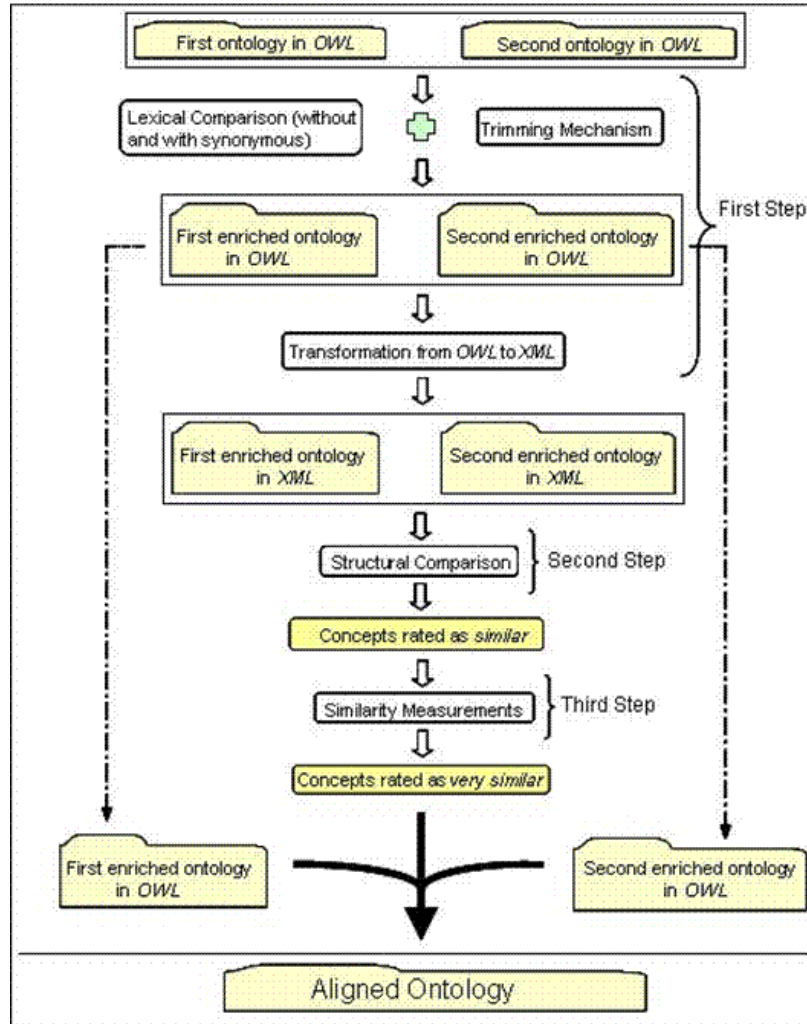


Fig. 2. CATO ontology alignment strategy

### 3.3.1 First Step: Lexical Comparison

The goal of this step is to identify lexically equivalent concepts. We assume that lexically equivalent concepts are also semantically equivalent in the domain of discourse under consideration, an assumption which is not always warranted.

Each concept label in the first ontology is compared to every concept label present in the second one, using lexical similarity as the criteria. Filters are used to normalize the labels to a canonical format: (i) If the concept is a noun, the canonical format is

the singular masculine declination; (ii) if the concept they represent is a verb, the canonical format is its infinitive. Besides using the label itself, synonyms are also used. The use of synonyms enriches the comparison process because it provides more refined information. For example, the case study concepts "TechReport" and "TechnicalReport" were identified as synonyms in our database.

Lexical similarity alone is not enough to assume that concepts are semantically compatible. We also investigate whether their ancestors share lexical similarity. Figure 3 shows an example of a situation in which, despite the fact that CATO identified the concept "Conference", present in both ontologies, as synonyms, alignment was not made because the concepts were not structurally compatible. Indeed, in the first ontology, the ancestor of "Conference" is the root concept whereas, in the second ontology, the ancestor of the "Conference" concept is "Event".

It is important to note that the alignment strategy in this step is restricted to concepts and instances of the ontology. We are not considering properties at this time. A concept instance is represented by a pair name and namespace in OWL.

As a result of the first stage of the proposed strategy, the original ontologies are enriched with links that relate concepts identified as lexically equivalent.

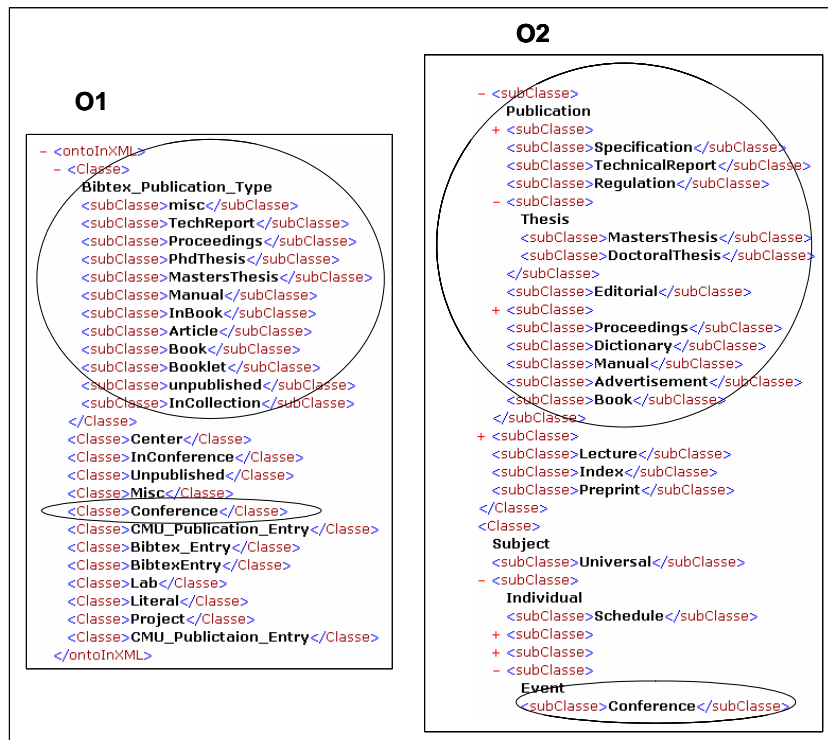


Fig. 3. Compared Ontologies

### 3.3.2 Second Step: Structural Comparison Using *TreeDiff*

Comparison at this stage is based on the subsumption relationship that holds among ontology concepts. Ontology properties and restrictions are not taken into consideration. Our approach is thus more restricted than the one proposed in [21], that analyses the ontologies as graphs, taking into consideration both taxonomic and non taxonomic relationships among concepts.

Because we only consider lexical and structural relationships in our analysis, we are able to make use of well-known tree comparison algorithms. We are currently using the *TreeDiff* [16] implementation available at [29]. Our choice was based on its ability to identify structural similarities between trees in reasonable time.

The goal of the *TreeDiff* algorithm is to identify the largest common substructure between trees, described using the DOM (*Document Object Model*) model. This algorithm was first proposed to help detect the steps, including renaming, removing and addition of tree nodes, necessary to migrate from one tree to another (both trees are the inputs to the algorithm).

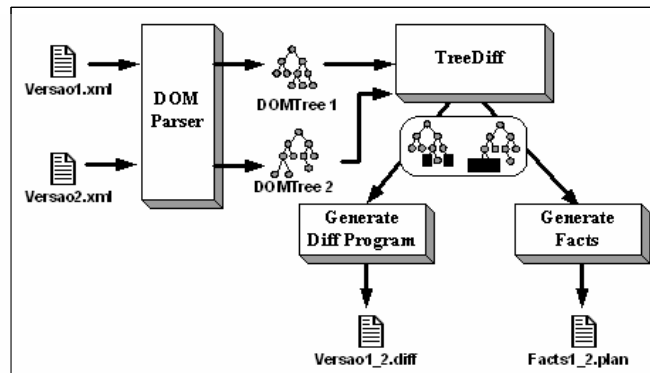


Fig. 4. The *TreeDiff* algorithm's entries and exits [29]

The result of the *Tree Diff* algorithm is the detection of concept equivalence groups. They are represented as subtrees of the enriched ontologies. Concepts that belong to such groups are compared in order to identify if lexically equivalent pairs can also be identified among the ancestors and descendants of the original pair. Differently from the first step, where we based our analysis and compared concepts that were directly related to one another, we are now considering the structural vicinity of concepts. Every concept in the equivalence group is investigated in order to determine lexically equivalent pairs, number of matching sons, number of synonymous concepts in the sub-trees, available from the previous step, and ancestor equivalence. An example of an equivalence group detected in the case study is illustrated by the ovals in Figure 3. Note that both ontologies in the case study possess a concept named "Proceedings". Additionally, in both cases, their super-concept contains the word *Publication* in their label. The equivalence group is thus identified by *CATO*. All concepts in the equivalence group are then compared.

The concepts "Book" and "Manual", identically labeled in both ontologies, have a lexically equivalent super-class ("Bibtex\_Publication\_Type" in O1 and "Publication" in O2), and are thus classified as equivalent.

Because of structural similarity (a single concept, i.e., a concept without sub-concepts, in O1 and a single sub-concept in O2) and lexical equivalence between the concepts labeled "Conference", another equivalence group, represented by the inferior circles in Figure 3, is also identified.

The "MastersThesis" and "Article" concepts, despite having the same the label for both ontologies, were not rated equivalent because of structural differences. The concepts "Proceedings", "Book", "Manual" and "Conference" fulfilled both requirements and were thus recognized as equivalent.

### 3.3.3. Third Step: Fine Adjustments based on Similarity Measurements

The third and last step is based on similarity measurements. Concepts are rated as very similar or little similar based on pre-defined similarity thresholds. We only align concepts that were both classified as lexically equivalent in the second step, and thus rated very similar. Thus the similarity measurement is the deciding factor responsible for fine tuning our strategy. We adapted the similarity measurement strategies proposed in [29, 30].

This is the case of concepts "Conference", "Proceedings", "Book" and "Manual", from the case study. Those concepts were rated equivalent during the second step (Note that they are part of the equivalence group illustrated in Figure 3). Their similarity level is calculated in the present step. Table I depicts the results. Please note that the concepts "Bibtex\_Publication\_Type" and "Publication", although considered equivalent at previous steps, did not achieve the necessary similarity percentage and, therefore, were not aligned.

The final ontology, showed in Figure 5, will provide a common understanding of the semantics represented by the two input ontologies. This representation can now be shared by software agents searching for information, seeking to discover or to compose with Web services [19].

**Table 1.** Similarity Percentages for concepts in the equivalence group illustrated in Figure 3

Similarities Level:		
Bibtex_Publication_Type -> Publication	***	Similarity Level: 23.076923%
Conference -> Conference	***	Similarity Level: 100.0%
Proceedings -> Proceedings	***	Similarity Level: 100.0%
Book -> Book	***	Similarity Level: 100.0%
Manual -> Manual	***	Similarity Level: 100.0%

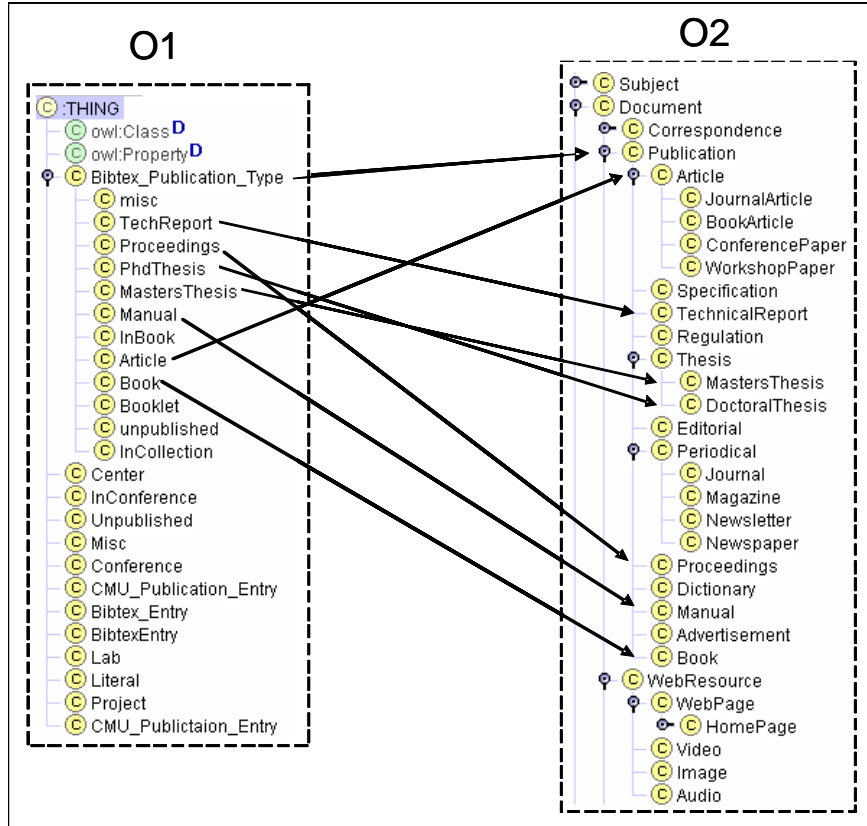


Fig. 5. Aligned concepts in both ontologies

## 4. Discussion

### 4.1 Alignment Strategy

In order to guarantee the desired response time and discard user intervention, some commitments had to be made. To guarantee reasonable performance, we limited our approach to lexical and structural comparisons. Much richer analysis could be performed if additional information was used, e.g. restrictions (slots) as it is done in both the Chimaera and Prompt approaches [6, 21].

For the sake of efficiency, we are only taking into consideration syntactical information, i.e., lexical and structural equivalence, in the proposed alignment strategy.

However, this limitation of the strategy can be overcome by the adaptation of the second step to take into consideration other ontology primitives, such as properties (the strategy could work with graphs instead of trees) and axioms. For sure this adaptation will increase the total computation time because of the added complexity.

We also implemented our strategy purely sequentially, without the possibility of feedback. Because every step of the strategy refines the previous one, better results could be achieved if manual feedback was allowed.

Furthermore, our alignment strategy is very conservative, in that it discards doubtful matches to preserve reliability. The worst case scenario in terms of completeness is not being able to align any concept. This happens when the input ontologies are from disjoint domains. The worst case scenario in terms of inconsistency is aligning two concepts that have identical labels, but are semantically different. This would only happen if, and only if, both shared identical labels and possessed a great deal of structural similarity, i.e., lexically equivalent concepts as descendants and/or ancestors.

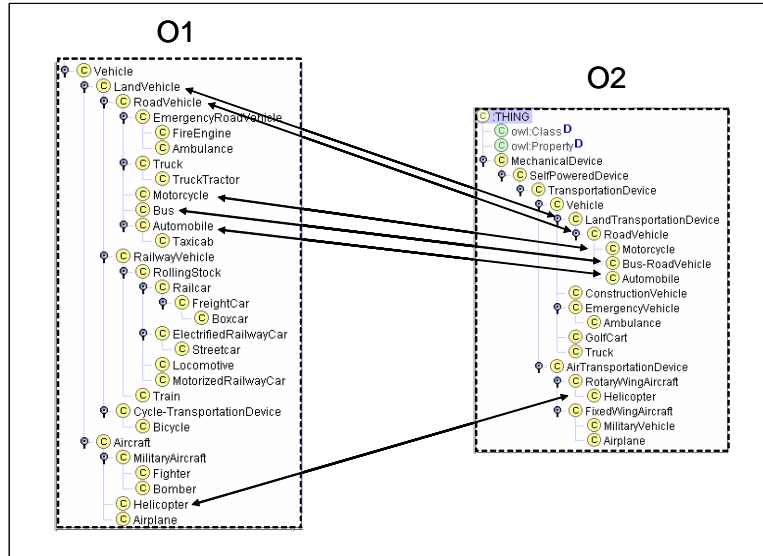
## 4.2 Implementation issues

CATO was fully implemented in Java and relies on the use of the JENA API. The use of the API helped us focus on the alignment process, for it made ontology manipulation transparent. JENA reads and filters information from the tags of files written in an ontology language and transforms it to an abstract data model in which ontological concepts can be manipulated as objects.

During the construction of CATO some adjustments to the algorithms had to be made. In particular, the refinement of the algorithms used in the manipulation of equivalence groups (step 2 of the strategy) made a great impact in the alignment results. We experimented with two manipulation algorithms for the equivalence groups. In the first implementation, we alphabetically ordered the sons of each node of the equivalence group sub-tree. In the second implementation, we maintained the original order in which the concepts appeared in the ontology. For pairs of ontologies that made use of identical labels, the use of the alphabetically ordered structural comparison files brought significantly better results. This was due to the fact that, after the alphabetical sort, the concepts present in an equivalence group will be more closely located in the structure. However, ontologies that make use of identical labels are rarely the case in practice. We tested the performance of both files on ontologies that had few identically labeled concepts. The ordered file did not bring differences in the results and there were some cases when it made the results worse than using the unordered file.

The *TreeDiff* algorithm, used in the second step of the strategy, is unidirectional in the sense that its goal is to determine the transformation needed to go from the first input tree to the second input tree. We are currently experimenting with double runs of the algorithm in which we invert the order of the input (today we are using the biggest ontology as the first input). When compared, the results present some differences. We believe we can refine the results from this algorithm by providing the

combination of the two runs. Future plans include continuing validation of the approach by experimentation and the elaboration of more case studies.



**Fig. 6.** Additional case study involving partitions of the SUMO and OpenCyc ontologies

### 4.3 Additional case studies

We performed additional case studies using upper ontologies as input. Of course the alignment of upper ontologies is outside the scope of our proposal. We focus on the alignment of lightweight ontologies produced in the Semantic Web context. Our intent with those experiments was to test the robustness and performance of CATO when dealing with very large volumes of information. We were very satisfied with the results, since the largest pair of ontologies took no longer than a few minutes to process. We were very surprised with the number of concepts CATO was able to align in those cases. In Figure 6 we illustrate one such case study. We compared concepts related to the Means of Transportation domain from both the SUMO (Suggested Upper Merged Ontology) and OpenCyc (Open source version of the Cyc Technology) upper ontologies.

## 5. Conclusions

In this paper, we discussed the implementation of a software component responsible for the automatic taxonomical alignment of ontologies. Our strategy is based on the

application of well known software engineering strategies, such as lexical analysis, tree comparison and the use of similarity measurements, to the problem of ontology alignment. Motivated by the requirements of multi agent systems, we proposed an ontology alignment strategy and tool that produces an intermediate ontological representation that makes it possible for software agents searching for information to share common understanding over information available on the Web [31, 32 and 33].

Our perception is that the future is not in trying to obtain total alignment in ontologies. The effort involved is too great and may not be justifiable in the context in which the ontologies will operate. Among the problems are the duration of interaction between two applications: Do software agents have enough time to align the ontologies? Are the requirements so ephemeral in nature that it is cost effective to allow the interaction even in the presence of inconsistency? How much mismatch/inconsistency is tolerable? Are levels of similarity an acceptable measure? Is it possible to analyze the impact and the risks involved in tolerating inconsistency between ontological representations? Can we apply classical inconsistency handling approaches to ontologies, such as the one proposed in [34]?

We are convinced that the solutions to the ontology integration problem are intertwined with our abilities to tolerate and live with inconsistencies that, as put by Easterbrook and Chechnik, are "a fact of live" [35]. It is not an easy shift however, for we have been trained to strive for completeness, consistency and to avoid conflict.

## References

1. Berners-Lee, T.; Lassila, O. Hendler, J.: The Semantic Web. Scientific American, May 2001. Available at: <http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html>>. Accessed on November, 2004.
2. Fellbaum, C.; ed: WordNet: An electronic Lexical Database. Cambridge, MA . MIT Press, 1998.
3. Guha, R. V., D. B. Lenat, K. Pittman, D. Pratt, and M. Shepherd: Cyc: A Midterm Report. Communications of the ACM Vol.33 , No. 8 - August, 1990.
4. Hendler, J.: Agents and the Semantic Web. IEEE Intelligent Systems. March/April, pp.30-37, 2001.
5. Bechhofer, S., Ian Horrocks, Carole Goble, Robert Stevens: OilEd: a Reason-able Ontology Editor for the Semantic Web. Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence, September 19-21, Vienna. Springer-Verlag LNAI Vol. 2174, pp. 396-408, 2001.
6. McGuinness, D.; Fikes, R.; Rice, J.; Wilder, S.: An Environment for Merging and Testing Large Ontologies. Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR-2000), Breckenridge, Colorado, April 12-15, San Francisco: Morgan Kaufmann, pp. 483-493, 2002.
7. Maedche, A.: Ontology Learning for the Sematic Web. Kluwer Academic Publishers, 2002.
8. Fensel, D.; Wahlster, W.; Berners-Lee, T.; editors: Spinning the Semantic Web. MIT Press, Cambridge Massachusetts, 2003.
9. Gómez-Peréz, A.; Fernández-Lopéz, Corcho, O.: Ontology Engineering. Springer Verlag, 2004.

10. Ushold, M; Gruninger, M.: Ontologies: Principles, Methods and Applications. Knowledge Engineering Review. Vol 11 No.2 - 1996.
11. Guarino, N.: Formal Ontology and information systems. In Proceedings of the FOIS'98 – Formal Ontology in Information Systems, Trento – 1998.
12. Noy, N.; McGuinness, D.: Ontology Development 101 – A guide to creating your first ontology. KSL Technical Report, Stanford University, 2001.
13. Booch, G.; Rumbaugh, J.; Jacobson, I.: The Unified Modeling Language user guide. Addison Wesley - 1999.
14. Yu, E.: Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering. Proceedings of the Third International Symposium on Requirements Engineering - RE97. IEEE Computer Society Press, pp.226-235, 1997.
15. Sowa, J. F.: Knowledge Representation: Logical, Philosophical and Computational Foundations. Brooks/Cole Books, Pacific Grove, CA, 2000.
16. Wang, J.: An Algorithm for Finding the Largest Approximately Common Substructures of Two Trees. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 20, Number 8, pp. 889-895, 1998.
17. TAI, K.,C.: The tree-to-tree correction problem. Journal of the ACM, 26(3), pp. 422-433, 1979.
18. M. Wooldridge, N. R. Jennings, and D. Kinny: A methodology for agent-oriented analysis and design. In O. Etzioni, J. P. Muller, and J. Bradshaw, editors, Agents '99: Proceedings of the Third International Conference on Autonomous Agents, Seattle, WA, May 1999.
19. Williams, A.B.: Learning to Share Meaning in a Multi-Agent System. Journal of Autonomous Agents and Multi-Agent Systems, Vol. 8, No. 2, 165-193, March 2004.
20. Noy, N. F., Musen, M. A.: SMART: Automated Support for Ontology Merging and Alignment. Workshop on Knowledge Acquisition, Modeling, and Management, Banff, Alberta, Canada, 1999.
21. Noy, N. F., Musen, M. A.: The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping. International Journal of Human-Computer Studies, 2003.
22. Pinto, S.H.; Gómez-Peréz, A.; Martins, J.P.: Some Issues on Ontology Integration. In: Workshop on Ontologies and Problems Solving Methods: Lessons Learned and Future Trends. Proceedings of the Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends (IJCAI99), 1999.
23. Doan, A., et. al.: Learning to match ontologies on the Semantic Web. In: The VLDB Journal — The International Journal on Very Large Data Bases, Volume 12, Issue 4, 2003. ISSN: 1066-8888. pp. 303-319, 2003.
24. Farquhar, A. Fikes, R.; Rice, J.: The Ontolingua Server a Tool for Collaborative Ontology Construction. Proceedings of the Tenth Knowledge Acquisition for Knowledge Base Systems Workshop, Banff, Canada, 1996.
25. Jena, the Semantic Web Framework, Available at: <<http://jena.sourceforge.net/>>. Accessed on November, 2004.
26. CMU RI Publications. Available at: <<http://www.daml.ri.cmu.edu/ont/homework/cmu-ri-publications-ont.daml/>>. Accessed on November, 2004.
27. Agent Transaction Language for Advertising Services. Available at: <<http://www.daml.ri.cmu.edu/>>. Accessed on November, 2004.
28. Mondeca SA, A Semantic Knowledge Company. Available at: <<http://www.mondeca.com/>>. Accessed on November, 2004.
29. Bergmann, U.: "Evolução de Cenários Através de um Mecanismo de Rastreamento Baseado em Transformações". PhD Thesis of the Department of Informatics of PUC-Rio, 2002.
30. Alexander Maedche and Steffen Staab: Comparing Ontologies Similarity Measures and a Comparison Study. Institute AIFB, University of Karlsruhe, Internal Report, 2001.

31. Williams, A.B., Padmanabhan, A., Blake, M.B.: Local Consensus Ontologies for B2B-Oriented Service Discovery. Second International Joint Conference on Autonomous Agents and Multi-Agent Systems, Melbourne, Australia, July 14-18, 2003.
32. Haendchen, F., A.; Staa, A.v.; Lucena, C.J.P: A Component-Based Model for Building Reliable Multi-Agent Systems. In Proceedings of 28th SEW - NASA/IEEE Software Engineering Workshop, Greenbelt, MD, IEEE Computer Society Press, Los Alamitos, CA, 2003.
33. Breitman, K.K., Haendchen, A.F., Staa, A., Haeusler, H.: Using Ontologies to Formalize Services Specifications in Multi-Agent Systems - Third NASA - Goddard/ IEEE Workshop FAABS III - Formal Approaches to Agent-Based Systems - Greenbelt, MA - April, 2004.
34. Nuseibeh, B.; Easterbrook, S.; Russo, A.: Leverage Inconsistency in Software Development Computer. - Vol 33 No. 4 - April 2000 - pp. 24-29, 2000.
35. Easterbrook, S.; Chechik, M. - 2nd International Workshop on Living with Inconsistency – Summary, IEEE, 2001.
36. D. McGuinness, R. Fikes, J. Rice, and S. Wilder: The Chimaera Ontology Environment. In Proceedings of the 17th National Conference on Artificial Intelligence (AAAI), 2000.