6
Referências Bibliográficas

ARTICULATION SERVICE. ***Articulation Service****, Version 0.2*. Disponível em: <http://codip.grci.com/Tools/ArtiServicePage.html>. Acesso em Junho de 2004.

ATLAS. ***Agent Transaction Language for Advertising Services***. Disponível em: <http://www.daml.ri.cmu.edu>. Acesso em Junho de 2004.

BERGMANN, U. **Evolução de Cenários Através de um Mecanismo de Rastreamento Baseado em Transformações**. Tese de Doutorado do Departamento de Informática da PUC-Rio, 2002.

BERNERS-LEE, T. B. ***Rules and Facts: Inference engines vs Web.*** Nota [On-line]. Última modificação: Janeiro de 2000a. Disponível em: <http://www.w3.org/DesignIssues/Rules.html>. Acesso em Junho de 2004.

BERNERS-LEE, T. B. ***Building the future***. Disponível em: <http://www.w3.org/2000/Talks/0906-xmlweb-tbl/slide9-6.html>. Apresentação titulada: **XML and the Web**, realizada no ano de 2000b. Acesso em Junho de 2004.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. *The Semantic Web.* ***Scientific American*** 284(5), p. 34-43, Maio de 2001. Disponível em: <http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21>. Acesso em Junho de 2004.

BOUQUET, P.; SERAFINI, L.; ZANOBINI, S. *Semantic Coordination: A New Approach and an Application.* In: INTERNATIONAL SEMANTIC WEB CONFERENCE, 2.2003, Florida. **Anais do International Semantic Web Conference**. Florida, 2003. p. 130-143.

BREITMAN, K. K.; Leite, J.C.S.P. *Lexicon Based Ontology Construction.* In: C. Lucena, C.; Garcia, A.; Romanovsky, A.; Castro, J.; Alencar, P.S.C. (Eds), **Software Engineering for Multi-Agent Systems II**. Springer-Verlag, LNCS 2940, Janeiro, 2004, ISBN: 3-540-21182-9, p. 19-34.

BRICKLEY, D. ***Resource Description Framework (RDF) Vocabulary Description Language 1.0: RDF Schema***. Recomendação da W3C a partir de 10 de Fevereiro de 2004. Disponível em: <http://www.w3.org/TR/rdf-schema/>. Acesso em Junho de 2004.

BURANARACH, M. ***The Foundation for Semantic Interoperability on the World Wide Web***. *PhD Thesis. Department of Information Science and Telecommunications School of Information Sciences, University of Pittsburgh*, Novembro, 2001.

CMU. ***Carnegie Mellon University***. Disponível em: <http://www.cmu.edu/>. Acesso em Junho de 2004a.

CMU. ***The Robotics Institute***. Disponível em: <http://www.ri.cmu.edu/>. Acesso em Junho de 2004b.

CMU. ***The Intelligent Software Agents Lab***. Disponível em: <http://www-2.cs.cmu.edu/~softagents/>. Acesso em Junho de 2004c.

CMU. ***CMU RI Publications Ontology***, *Id: cmu-ri-publications-ont.daml, version 0.1, 2001/08/27*. Disponível em <http://www.daml.ri.cmu.edu/ont/homework/cmu-ri-publications-ont.daml>. Acesso em Junho de 2004d.

CMU. ***CMU RI Employment Categories Ontology***. *Id: cmu-ri-employmenttypes-ont.daml, version 0.1, 2001/08/27*. Disponível em <http://www.daml.ri.cmu.edu/ont/homework/cmu-ri-employmenttypes-ont.daml>. Acesso em Junho de 2004e.

CONNOLLY, D.; HARMELEN, F. v.; HORROCKS, I.; MCGUINNESS, D. L.; PATEL-SCHNEIDER, P. F.; STEIN, L. A. ***DAML+OIL (March 2001) Reference Description***. 2001. Disponível em: <http://www.w3.org/TR/daml+oil-reference>. Acesso em Junho de 2004.

DAML. ***The DARPA Agent Markup Language Homepage***. Disponível em: <http://www.daml.org/>. Acesso em Junho de 2004.

DAML ONTOLOGY LIBRARY. *Version from April, 2004*. Disponível em: <http://www.daml.org/ontologies/>. Acesso em Junho de 2004.

DAML TOOLS. *Version* 1.257. Disponível em: <http://www.daml.org/tools/>. Acesso em Junho de 2004.

DEAN, M.; SCHREIBER, G.; BECHHOFER, S.; HARMELEN, F. v.; HENDLER, J.; HORROCKS, I.; MCGUINNESS, D. L.; PATEL-SCHNEIDER, P. F.; STEIN, L. A. ***OWL Web Ontology Language Reference***. Recomendação da W3C a partir de 10 de Fevereiro de 2004. Disponível em: <http://www.w3.org/TR/owl-ref/>. Acesso em Maio de 2004a.

DEAN. ***OWL Web Ontology Language Use Cases and Requirements***. Disponível em: <http://www.w3.org/TR/webont-req/>. Acesso em Junho de 2004b.

DECKER, S., MELNIK, S.; HARMELEN, F. v.; FENSEL, D.; KLEIN, M.; BROEKSTRA, J.; ERDMANN, M.; HORROCKS, I. ***The Semantic Web: The roles of XML and RDF***. IEEE Internet Computing, 4, p. 63-73, 2000. Disponível em: <http://www.ida.liu.se/~asmpa/courses/sweb/rdf/roles_of_xml_and_rdf.pdf>. Acesso em Julho de 2004.

DFKI. ***German Research Center for Artificial Intelligence***. Disponível em: <http://www.dfki.de/relaunch/>. Acesso em Junho de 2004.

DOAN, A., MADHAVAN, J.; DHAMANKAR, R.; DOMINGOS, P.; HALEVY, A. *Learning to match ontologies on the Semantic Web. In:* ***The VLDB Journal — The International Journal on Very Large Data Bases***, v. 12, n. 4, p. 303–319, ISSN:1066-8888, Novembro de 2003.

DOM. ***W3C Document Object Model (DOM)***. Disponível em: <http://www.w3.org/DOM/>. Acesso em Junho de 2004.

FELICÍSSIMO, C. H; SILVA, L. F. S.; BREITMAN, K. K.; LEITE, J. C. S. P. Geração de Ontologias subsidiada pela Engenharia de Requisitos. In: **WORKSHOP EM ENGENHARIA DE REQUISITOS**, 6. 2003a, Piracicaba. Anais do Workshop em Engenharia de Requisitos. Piracicaba, São Paulo, 2003a. p. 255-269.

FELICÍSSIMO, C. H. **Projeto de Programação para o Curso de Mestrado** do Programa de Pós-Graduação do Departamento de Informática da PUC-Rio. Dezembro de 2003b.

FELLBAUM, C. ***WordNet: An electronic Lexical Database***. Cambridge, MA, MIT Press, 1998. Versão para *download* do WordNet disponível em: <http://www.cogsci.princeton.edu/~wn/>. Acesso em Julho de 2004.

FELLBAUM. ***Person in WordNet.*** Disponível em: <http://xmlns.com/wordnet/1.6/Person>. Acesso em Junho de 2004.

FENSEL, D. ***Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce***. Springer Verlag, 2001.

GOMEZ-PEREZ, A.; FERNÁNDEZ-LÓPEZ, M.; CORCHO, O. ***Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web***. Series: Advanced Information and Knowledge Processing, 2004, XII, 403 p. 159 illus., Hardcover ISBN: 1-85233-551-3.

GOOGLE. ***Google Directory, Published Ontologies***. Disponível em: <http://directory.google.com/Top/Reference/Knowledge_Management/Knowledge_Representation/Ontologies/Published_Ontologies/>. Acesso em Junho de 2004.

GRUBER, T. R. ***A translation approach to portable ontology specifications***. Knowledge Acquisition, v. 5, n. 2, p. 199-220, ISSN:1042-8143, Junho de 1993.

HAENDCHEN, F. A.; STAA, A. v.; LUCENA, C. P. *A Component-Based Model for Building Reliable Multi-Agent Systems*. In: SEW - NASA/IEEE SOFTWARE ENGINEERING WORKSHOP, 28. 2003, Los Alamitos. **Anais do SEW - NASA/IEEE Software Engineering Workshop**. Los Alamitos, 2003.

HENDLER, J. ***Agents and the Semantic Web***. In: IEEE Intelligent Systems, v. 16, n. 2, Março/Abril, 2001, p.30-37.

HORROCKS, I.; PATEL-SCHNEIDER, P. F.; HARMELEN, F. v. From SHIQ and RDF to OWL: *The Making of a Web Ontology Language. In:* ***Journal of Web Semantics,*** 1(1), 2003. p. 7-26.

HP. ***HP Labs Semantic Web Research***. Disponível em: <http://www.hpl.hp.com/semweb/>. Acesso em Junho de 2004.

HTML. ***HyperText Markup Language (HTML) Home Page***. Disponível em: <http://www.w3.org/MarkUp/>. Acesso em Junho de 2004.

IST-PRIZE. ***The European IST Prize, Information Society Technologies***. Disponível em: <http://www.ist-prize.org/>. Acesso em Junho de 2004.

JAVA. ***The JavaTM Virtual Machine Specification***. *Second Edition*. LINDHOLM, T., YELLIN, F. 1999. Disponível em: <http://java.sun.com/docs/books/vmspec/2nd-edition/html/VMSpecTOC.doc.html>. Acesso em Junho de 2004.

JAVA. ***The Java Language Specification***. *Second Edition*. GOSLING, J.; JOY, B.; STEELE, G.; BRACHA, G. 2000. Disponível em: <http://java.sun.com/docs/books/jls/second_edition/html/j.title.doc.html>. Acesso em Junho de 2004.

JENA. ***Jena - A Semantic Web Framework for Java***. Disponível em: <http://jena.sourceforge.net/>. Acesso em Junho de 2004a.

JENA. ***Jena 2 Ontology API***. Disponível em: <http://jena.sourceforge.net/ontology/>. Acesso em Junho de 2004b.

LEITE, J.C.S.P., FRANCO, A.P.M. *A Strategy for Conceptual Model Acquisition*. In: INTERNATIONAL SYMPOSIUM ON REQUIREMENTS ENGINEERING**,** 1. 1993. **Anais do International Symposium on Requirements Engineering**, IEEE Computer Society Press, 1993. p. 243-246.

MAEDCHE, A. S. ***Discovering Conceptual Relations from Text***. *Techical Report 400, University of Karlsruhe, Institute AIFB, 76128 Karlsruhe, Germany*. Fevereiro de 2000. Disponível em <http://www.aifb.uni-karlsruhe.de/Publikationen/showPublikationen?id_db=50> Acesso em Maio de 2004.

MAEDCHE, A. ***Ontology Learning for the Sematic Web***. Livro, Kluwer Academic Publishers, 2002, ISBN: 0792376560.

MAGALHÃES, J. **Um Framework Multi-Agentes para Busca e Flexibilização de Algoritmos de Classsificação de Documentos**. Dissertação de Mestrado, PUC-Rio, 2002.

MILLER, E.; MANOLA, F.; MCBRIDE, B. ***Resource Description Framework (RDF) Primer***. Recomendação da W3C a partir de 10 de Fevereiro de 2004. Disponível em: <http://www.w3.org/TR/rdf-primer/>. Acesso em Junho de 2004.

MONDECA. ***A Semantic Knowledge Company***. Disponível em: <http://www.mondeca.com>. Acesso em Junho de 2004a.

MONDECA. ***ITM - Intelligent Topic Manager***. Disponível em: <http://www.mondeca.com/technologie.htm>. Acesso em Junho de 2004b.

MONDECA. ***General University Ontology****, version 1.2*. Disponível em: <http://www.mondeca.com/owl/moses/univ.owl>. Acesso em Junho de 2004c.

MOREIRA, M. M. **Integração Semântica de Sistemas de Informação**. Dissertação de Mestrado, PUC-Rio, 2003.

NOKIA. ***Nokia Research Center***. Disponível em: <http://www.nokia.com/nokia/0,8764,50249,00.html>. Acesso em Junho de 2004.

NOY, N. F., MUSEN, M. A. *SMART: Automated Support for Ontology Merging and Alignment*. In: WORKSHOP ON KNOWLEDGE ACQUISITION, MODELING AND MANAGEMENT, 12. 1999a, Banff. **Anais do Workshop on Knowledge Acquisition, Modeling and Management**. Banff, 1999a. Disponível como relatório técnico da SMI em <http://smi-web.stanford.edu/pubs/SMI_Abstracts/SMI-1999-0813.html>. Acesso em Julho de 2004.

NOY, N. F. ***Presentation: Ontologies and tools***. 1999b. Disponível em: <http://protege.stanford.edu/publications/OntologiesAndTools/OntologiesAndTools.ppt>. Acesso em Junho de 2004.

NOY, N. F., MUSEN, M. A. *Anchor-PROMPT: Using Non-Local Context for Semantic Matching*. In: WORKSHOP ON ONTOLOGIES AND INFORMATION SHARING AT THE INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 17. 2001a, Seattle. Anais do **Workshop on Ontologies and Information Sharing at the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)**, Seattle, 2001a. Disponível como relatório técnico da SMI em <http://smi-web.stanford.edu/pubs/SMI_Abstracts/SMI-2001-0889.html>. Acesso em Julho de 2004.

NOY, N.; MCGUINNESS, D. ***Ontology Development 101 – A guide to creating your first ontology***. *KSL Technical Report, Standford University, 2001b.* Disponível em: <http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html>. Acesso em Junho de 2004.

NOY, N. F., MUSEN, M. A. *The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping*. In: **International Journal of Human-Computer Studies**, 59/6. p. 983-1024. 2003. Disponível como relatório técnico da SMI em <http://smi-web.stanford.edu/pubs/SMI_Abstracts/SMI-2003-0973.html>. Acesso em Julho de 2004.

NUSEIBEH, B.; EASTERBROOK, S.; RUSSO, A. ***Leveraging Inconsistency in Software Development***. In: IEEE Computer Society Press, v. 33, n. 4, 2000. p. 26-29.

OILED. ***An Ontology Editor***. Disponível em: <http://oiled.man.ac.uk/>. Acesso em Junho de 2004.

OMG. ***Object Management Group – Agent Platform Special Interest Group: Agent Technology – Green Paper****, Version 1.0, 2000*. Disponível em: <http://www.objs.com/agent/agents_Green_Paper_v100.doc>. Acesso em Junho de 2004.

ONTOMERGE. ***OntoMerge - Ontology Translation by Merging Ontologies***. Disponível em: <http://onto.cs.yale.edu:4040/ontoMerge.html>. Acesso em Junho de 2004.

OPEN SOURCE. ***Open Source Initiative - OSI***. Disponível em: <http://www.opensource.org/>. Acesso em Junho de 2004.

OPENCYC. ***Open source version of the Cyc technology***. Disponível em: <http://www.opencyc.org/>. Acesso em Junho de 2004.

PINTO, S. H.; GOMEZ-PEREZ, A. G.; MARTINS, J. P. *Some Issues on Ontology Integration*. In: WORKSHOP ON ONTOLOGIES AND PROBLEM SOLVING METHODS: LESSONS LEARNED AND FUTURE TRENDS. 1999. **Anais do *Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends* (IJCAI99's)**, 1999.

PROTEGE. ***Protégé Project***. Disponível em: <http://protege.stanford.edu/>. Acesso em Junho de 2004a.

PROTEGE. ***Protégé Contributions Library***. Disponível em: <http://protege.stanford.edu/plugins.html/>. Acesso em Junho de 2004b.

PROTEGE. ***Protégé PROMPT Plug-In***. Disponível em: <http://protege.stanford.edu/plugins/prompt/prompt.html/>. Acesso em Junho de 2004c.

RESNIK, P. *Knowledge maintenance: The state of the art.* ***In Proceedings of IJCAI***, p. 448-453, Montreal, Canada, 1995.

RICHARDSON, R.; SMEATON, A. F.; MURPHY, J. ***Using WordNet as Knowledge Base for Measuring Semantic Similarity between Words***. *Technical Report CA-1294, Dublin City University, School of Computer Applications, 1994.*

SCHEMAWEB. ***Browse SchemaWeb Directory***. Disponível em: <http://www.schemaweb.info/schema/BrowseSchema.aspx>. Acesso em Junho de 2004a.

SCHEMAWEB. ***SchemaWeb - Schema Details :: Wordnet Person***. Disponível em: <http://www.schemaweb.info/schema/SchemaDetails.aspx?id=25>. Acesso em Junho de 2004b.

SEMANTICWEB. ***Semantic Web Activity***. Disponível em: <http://www.w3.org/2001/sw>. Acesso em Junho de 2004.

SIKORA, Z. M. **<JAVA> GUIA PRÁTICO PARA PROGRAMADORES.** Editora Campus Ltda. 2003.

SMITH, M. K.; WELTY, C.; MCGUINNESS, D. L. ***OWL Web Ontology Language Guide***. Recomendação da W3C a partir de 10 de Fevereiro de 2004. Disponível em: <http://www.w3.org/TR/owl-guide/>. Acesso em Maio de 2004.

SOWA, J. ***Ontology***. Última atualização do Web Site em Julho de 2003. Disponível em: <http://www.jfsowa.com/ontology/index.htm>. Acesso em Junho de 2004.

SUMO. ***Suggested Upper Merged Ontology***. Disponível em: <http://ontology.teknowledge.com/>. Acesso em Junho de 2004.

SUOWG. ***Standard Upper Ontology Working Group***. Disponível em: <http://suo.ieee.org/>. Acesso em Junho de 2004.

SWI-Prolog. ***What is SWI-Prolog?***. Disponível em: <http://www.swi-prolog.org/>. Acesso em Junho de 2004.

TAI, K.,C. *The tree-to-tree correction problem.* ***Journal of the ACM***, v. 26, n. 3, 1979, p. 422-433.

TEKNOWLEDGE. ***An Ontology Mapping Tool from Teknowledge Company.*** Disponível em: <http://einstein.teknowledge.com:8080/DAML/DAML_register.jsp?fileType=.zip&fileName=ont_mapping.zip>. Acesso em Junho de 2004.

TREEMATCHER. ***TreeMatcher toolkit for comparing two ordered or unordered trees***. Disponível em: <http://www.cis.njit.edu/~discdb/treematcher.html>. Acesso em Junho de 2004.

USCHOLD, M. ***Building Ontologies: Towards a Unified Methodology***. In: ANNUAL CONFERENCE OF EXPERTS SYSTEMS, 16. 1996, Cambridge. Anais do Conference of Experts Systems. Cambridge, 1996.

XML. ***Extensible Markup Language (XML)***. Disponível em: <http://www.w3.org/XML/>. Acesso em Junho de 2004.

W3CSEMANTICWEB. ***W3C Semantic Web Activity***. Disponível em: <http://www.w3.org/2001/sw/>. Acesso em Junho de 2004.

W3CWEBONTWORKINGGROUP. ***W3C Web Ontology (WebOnt) Working Group***. Disponível em: <http://www.w3.org/2001/sw/WebOnt/>. Acesso em Junho de 2004.

WANG, J. ***An Algorithm for Finding the Largest Approximately Common Substructures of Two Trees***, In: IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 20, n. 8. , 1998, p. 889-895.

Anexo A – Código em *OWL* das Ontologias do Primeiro Estudo de Caso

Os códigos-fonte das ontologias de entrada do CATO deste estudo de caso encontram-se nas próximas páginas. Os códigos-fonte das ontologias de saída, dos módulos sem e com ordenação alfabética, podem ser obtidos nos seguintes endereços eletrônicos:

Ontologia de saída do módulo sem ordenação alfabética

http://www.inf.puc-rio.br/~cfelicissimo/frstSolCombPropFinalFile.owl

Ontologia de saída do módulo com ordenação alfabética

http://www.inf.puc-rio.br/~cfelicissimo/frstSolCombPropFinalFileOd.owl

Primeira Ontologia de entrada

```
<rdf:RDF xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#">

<owl:Ontology rdf:about=""/>


  <owl:Class rdf:ID="Bibtex_Publication_Type"/>

  <owl:Class rdf:ID="misc">
    <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
  </owl:Class>

  <owl:Class rdf:ID="Center"/>

  <owl:Class rdf:ID="InConference">
    <owl:disjointWith>
      <owl:Class rdf:about="#TechReport"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Misc"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Unpublished"
       rdf:type="http://www.w3.org/2002/07/owl#Class"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#MastersThesis"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Manual"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Proceedings"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#PhdThesis"/>
    </owl:disjointWith>
  </owl:Class>

  <owl:Class rdf:ID="Misc">
    <owl:disjointWith>
      <owl:Class rdf:about="#TechReport"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Unpublished"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Proceedings"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#PhdThesis"/>
    </owl:disjointWith>
  </owl:Class>

  <owl:Class rdf:ID="MastersThesis">
    <owl:disjointWith>
      <owl:Class rdf:about="#TechReport"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Misc"/>
    <owl:disjointWith rdf:resource="#Unpublished"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Proceedings"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#PhdThesis"/>
    </owl:disjointWith>
    <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
  </owl:Class>

  <owl:Class rdf:ID="Conference">
    <owl:disjointWith>
      <owl:Class rdf:about="#TechReport"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Misc"/>
    <owl:disjointWith rdf:resource="#Unpublished"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#InBook"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#MastersThesis"/>
    <owl:disjointWith rdf:resource="#InConference"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Manual"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Proceedings"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#PhdThesis"/>
    </owl:disjointWith>
  </owl:Class>
```

```
<owl:Class rdf:ID="Article">
  <owl:disjointWith>
    <owl:Class rdf:about="#TechReport"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Conference"/>
  <owl:disjointWith rdf:resource="#Misc"/>
  <owl:disjointWith rdf:resource="#Unpublished"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#InBook"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#MastersThesis"/>
  <owl:disjointWith rdf:resource="#InConference"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Manual"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Proceedings"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#PhdThesis"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Book"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Booklet"/>
  </owl:disjointWith>
</owl:Class>

<owl:Class rdf:ID="CMU_Publication_Entry"/>

<owl:Class rdf:ID="TechReport">
  <owl:disjointWith rdf:resource="#Unpublished"/>
  <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
</owl:Class>

<owl:Class rdf:ID="Bibtex_Entry"/>

<owl:Class rdf:ID="InBook">
  <owl:disjointWith rdf:resource="#TechReport"/>
  <owl:disjointWith rdf:resource="#Misc"/>
  <owl:disjointWith rdf:resource="#Unpublished"/>
  <owl:disjointWith rdf:resource="#MastersThesis"/>
  <owl:disjointWith rdf:resource="#InConference"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Manual"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Proceedings"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#PhdThesis"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
</owl:Class>

<owl:Class rdf:ID="BibtexEntry"/>

<owl:Class rdf:ID="Booklet">
  <owl:disjointWith rdf:resource="#TechReport"/>
```

```
    <owl:disjointWith rdf:resource="#Conference"/>
    <owl:disjointWith rdf:resource="#Misc"/>
    <owl:disjointWith rdf:resource="#Unpublished"/>
    <owl:disjointWith rdf:resource="#InBook"/>
    <owl:disjointWith rdf:resource="#MastersThesis"/>
    <owl:disjointWith rdf:resource="#InConference"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Manual"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Proceedings"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#PhdThesis"/>
    </owl:disjointWith>
    <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
  </owl:Class>

  <owl:Class rdf:ID="Lab"/>

  <owl:Class rdf:ID="unpublished">
    <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
  </owl:Class>

  <owl:Class rdf:ID="Literal"/>

  <owl:Class rdf:ID="InCollection">
    <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
  </owl:Class>

  <owl:Class rdf:ID="PhdThesis">
    <owl:disjointWith rdf:resource="#TechReport"/>
    <owl:disjointWith rdf:resource="#Unpublished"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Proceedings"/>
    </owl:disjointWith>
    <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
  </owl:Class>

  <owl:Class rdf:ID="Project"/>

  <owl:Class rdf:ID="Book">
    <owl:disjointWith rdf:resource="#TechReport"/>
    <owl:disjointWith rdf:resource="#Conference"/>
    <owl:disjointWith rdf:resource="#Misc"/>
    <owl:disjointWith rdf:resource="#Unpublished"/>
    <owl:disjointWith rdf:resource="#InBook"/>
    <owl:disjointWith rdf:resource="#MastersThesis"/>
    <owl:disjointWith rdf:resource="#InConference"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Manual"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Proceedings"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#PhdThesis"/>
    <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
    <owl:disjointWith rdf:resource="#Booklet"/>
  </owl:Class>

  <owl:Class rdf:ID="CMU_Publictaion_Entry"/>
```

```
<owl:Class rdf:ID="Manual">
  <owl:disjointWith rdf:resource="#TechReport"/>
  <owl:disjointWith rdf:resource="#Misc"/>
  <owl:disjointWith rdf:resource="#Unpublished"/>
  <owl:disjointWith rdf:resource="#MastersThesis"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Proceedings"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#PhdThesis"/>
  <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
</owl:Class>

<owl:Class rdf:ID="Proceedings">
  <owl:disjointWith rdf:resource="#TechReport"/>
  <owl:disjointWith rdf:resource="#Unpublished"/>
  <rdfs:subClassOf rdf:resource="#Bibtex_Publication_Type"/>
</owl:Class>

<owl:ObjectProperty rdf:ID="hasBibtexEntry">
  <rdfs:range rdf:resource="#BibtexEntry"/>
  <rdfs:domain rdf:resource="#CMU_Publictaion_Entry"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasSeries">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasPages">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasOrganization">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasNote">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasPublication_Type">
  <rdfs:domain rdf:resource="#Bibtex_Entry"/>
  <rdfs:range rdf:resource="#Bibtex_Publication_Type"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasVolume">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasMonth">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasType">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasAuthor">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasURL">
  <rdfs:domain rdf:resource="#CMU_Publictaion_Entry"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasGrantID">
  <rdfs:domain rdf:resource="#CMU_Publictaion_Entry"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasSchool">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasChapter">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasYear">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasKey">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasAnnote">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasInstitution">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasJournal">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasNumber">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasAssociatedLabGroup">
  <rdfs:range rdf:resource="#Lab"/>
  <rdfs:domain rdf:resource="#CMU_Publictaion_Entry"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasPublisher">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasHowpublished">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasEditor">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasAddress">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasAssociatedProject">
  <rdfs:range rdf:resource="#Project"/>
  <rdfs:domain rdf:resource="#CMU_Publictaion_Entry"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasBooktitle">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasAssociatedCenter">
  <rdfs:domain rdf:resource="#CMU_Publictaion_Entry"/>
  <rdfs:range rdf:resource="#Center"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasEdition">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasSponsor">
  <rdfs:domain rdf:resource="#CMU_Publictaion_Entry"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasCiteKey">
  <rdfs:domain rdf:resource="#Bibtex_Entry"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasCrossref">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
  <rdfs:range rdf:resource="#Literal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasTitle">
  <rdfs:domain rdf:resource="#Bibtex_Publication_Type"/>
```

```
    <rdfs:range rdf:resource="#Literal"/>
  </owl:ObjectProperty>

</rdf:RDF>
```

Segunda Ontologia de entrada

```
<rdf:RDF xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

    <owl:Ontology dc:date="2003-10-20" dc:format="text/xml"
dc:language="en" dc:publisher="Mondeca"
rdf:about="http://www.mondeca.com/owl/moses/univ.owl">
        <owl:versionInfo>version 1.2</owl:versionInfo>
        <owl:priorVersion
rdf:resource="http://www.mondeca.com/owl/univ1_1.xml"/>
        <dc:title>General University Ontology</dc:title>
        <dc:creator>Bernard Vatant</dc:creator>
        <dc:description>General and University
Ontology</dc:description>
<dc:identifier>http://www.mondeca.com/owl/moses/univ.owl</dc:ident
ifier>
<dc:source>http://www.cs.umd.edu/projects/plus/DAML/onts/univ1.0.d
aml</dc:source>
    </owl:Ontology>

    <owl:Class rdf:ID="Director">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#AdministrativeStaff"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Discussion">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Correspondence"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="researchProject">
        <rdfs:subClassOf rdf:resource="#Association"
rdf:type="http://www.w3.org/2002/07/owl#Class"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#researchProjectSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#ResearchGroup"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#researchProjectObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Research"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Article">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Publication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="workAddress">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#workAddressSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Person"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#workAddressObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Address"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="AssociateProfessor">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Professor"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="orgAddress">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#orgAddressSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#orgAddressObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Address"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Individual">
        <owl:sameAs
rdf:resource="http://www.mondeca.com/owl/otm.xml#Individual"/>
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Subject"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="isLocated">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#isLocatedSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#isLocatedObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Location"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="PhysicalObject">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="Specification">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Publication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="communicator">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#communicatorSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Communication"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#communicatorObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Agent"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Lexicalization">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#occuring"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Subject"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#inDocument"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Document"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#form"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Lexical"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="WebPage">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#WebResource"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Email">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Correspondence"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="NonprofitOrganization">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Organization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Video">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#WebResource"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Minutes">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="listedCourse">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#listedCourseSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Schedule"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
```

```
                <owl:onProperty rdf:resource="#listedCourseObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Course"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Schedule">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="DocumentRepresentation">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Artifact"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Correspondence">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Professor">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Faculty"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Manuscript">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="TechnicalReport">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Publication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="advisor">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#advisorSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Student"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#advisorObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Professor"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="recipient">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#recipientSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
```

```
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Communication"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#recipientObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Agent"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Work">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Activity"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="researchInterest">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#researchInterestSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Person"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#researchInterestObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Research"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="authorOrg">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#authorOrgSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Document"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#authorOrgObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Recreation">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Activity"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="affiliateOf">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#affiliateOfSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#affiliateOfObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Person"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="participant">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#participantSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Event"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#participantObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Agent"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="ResearchAssistant">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Assistant"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Department">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#EducationOrganization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Guideline">
```

```
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Dean">
        <rdfs:subClassOf rdf:resource="#Professor"/>
        <rdfs:subClassOf>
            <owl:Class rdf:about="#AdministrativeStaff"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="AdministrativeStaff">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Employee"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="teachingAssistantOf">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#teachingAssistantOfSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#TeachingAssistant"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#teachingAssistantOfObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Course"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="affiliatedOrganization">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#affiliatedOrganizationSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#affiliatedOrganizationObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Institute">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#EducationOrganization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Subject">
        <owl:sameAs
rdf:resource="http://www.mondeca.com/owl/otm.xml#Subject"/>
    </owl:Class>
    <owl:Class rdf:ID="University">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#EducationOrganization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Program">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#EducationOrganization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Index">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="School">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#EducationOrganization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Regulation">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Publication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Event">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="Review">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Activity">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="MastersThesis">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Thesis"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="PostDoc">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Faculty"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="GraduateStudent">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Student"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="FullProfessor">
```

```
        <rdfs:subClassOf rdf:resource="#Professor"/>
    </owl:Class>
    <owl:Class rdf:ID="Image">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#WebResource"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Editorial">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Publication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Periodical">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Publication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="eventLocation">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#eventLocationSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Event"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#eventLocationObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Location"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="ElectronicDocument">
        <rdfs:subClassOf rdf:resource="#DocumentRepresentation"/>
    </owl:Class>
    <owl:Class rdf:ID="offersCourse">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#offersCourseSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#University"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#offersCourseObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Course"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="ResearchGroup">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#EducationOrganization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="ArtificialAgent">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Agent"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Abstract">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="WebResource">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Publication">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Comment">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Student">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Person"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Preprint">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Document"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="JournalArticle">
        <rdfs:subClassOf rdf:resource="#Article"/>
    </owl:Class>
    <owl:Class rdf:ID="Journal">
        <rdfs:subClassOf rdf:resource="#Periodical"/>
    </owl:Class>
    <owl:Class rdf:ID="mastersDegreeFrom">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#mastersDegreeFromSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Person"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
```

```
                <owl:onProperty
rdf:resource="#mastersDegreeFromObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#University"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Universal">
        <owl:sameAs
rdf:resource="http://www.mondeca.com/owl/otm.xml#Universal"/>
        <rdfs:subClassOf rdf:resource="#Subject"/>
    </owl:Class>
    <owl:Class rdf:ID="Location">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="OrganizationHomepage">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#HomePage"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="documentSubject">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#documentSubjectSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Document"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#documentSubjectObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Subject"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="borders">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#bordersSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Location"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#bordersObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Location"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Proceedings">
        <rdfs:subClassOf rdf:resource="#Publication"/>
    </owl:Class>
    <owl:Class rdf:ID="HomePage">
        <rdfs:subClassOf rdf:resource="#WebPage"/>
    </owl:Class>
    <owl:Class rdf:ID="PhoneCall">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Communication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Conference">
        <rdfs:subClassOf rdf:resource="#Event"/>
    </owl:Class>
    <owl:Class rdf:ID="Software">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Artifact"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Letter">
        <rdfs:subClassOf rdf:resource="#Correspondence"/>
    </owl:Class>
    <owl:Class rdf:ID="Document">
        <owl:sameAs
rdf:resource="http://www.mondeca.com/owl/otm.xml#Document"/>
    </owl:Class>
    <owl:Class rdf:ID="alumnus">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#alumnusSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#alumnusObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Person"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Speech">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Communication"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Postcard">
        <rdfs:subClassOf rdf:resource="#Correspondence"/>
    </owl:Class>
    <owl:Class rdf:ID="Research">
        <rdfs:subClassOf rdf:resource="#Work"/>
    </owl:Class>
    <owl:Class rdf:ID="publicationResearch">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
```

```
                <owl:onProperty
rdf:resource="#publicationResearchSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Publication"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#publicationResearchObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Research"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="containedIn">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#containedInSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Document"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#containedInObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Document"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="TeachingAssistant">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Assistant"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="performs">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#performsSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Person"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#performsObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Work"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Magazine">
        <rdfs:subClassOf rdf:resource="#Periodical"/>
    </owl:Class>
    <owl:Class rdf:ID="Lexical">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="encloses">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#enclosesSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Location"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#enclosesObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Location"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Person">
        <rdfs:subClassOf rdf:resource="#Individual"/>
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Agent"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Communication">
        <rdfs:subClassOf rdf:resource="#Event"/>
    </owl:Class>
    <owl:Class rdf:ID="Lecture">
        <rdfs:subClassOf rdf:resource="#Document"/>
    </owl:Class>
    <owl:Class rdf:ID="Dictionary">
        <rdfs:subClassOf rdf:resource="#Publication"/>
    </owl:Class>
    <owl:Class rdf:ID="BookArticle">
        <rdfs:subClassOf rdf:resource="#Article"/>
    </owl:Class>
    <owl:Class rdf:ID="ConferencePaper">
        <rdfs:subClassOf rdf:resource="#Article"/>
    </owl:Class>
    <owl:Class rdf:ID="engagesIn">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#engagesInSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Agent"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#engagesInObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Activity"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="PaperDocument">
        <rdfs:subClassOf rdf:resource="#DocumentRepresentation"/>
    </owl:Class>
    <owl:Class rdf:ID="Agent">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="Lecturer">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Faculty"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Manual">
        <rdfs:subClassOf rdf:resource="#Publication"/>
    </owl:Class>
    <owl:Class rdf:ID="Newsletter">
        <rdfs:subClassOf rdf:resource="#Periodical"/>
    </owl:Class>
    <owl:Class rdf:ID="Course">
        <rdfs:subClassOf rdf:resource="#Work"/>
    </owl:Class>
    <owl:Class rdf:ID="AssistantProfessor">
        <rdfs:subClassOf rdf:resource="#Professor"/>
    </owl:Class>
    <owl:Class rdf:ID="publisher">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#publisherSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Document"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#publisherObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Audio">
        <rdfs:subClassOf rdf:resource="#WebResource"/>
    </owl:Class>
    <owl:Class rdf:ID="worksFor">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#worksForSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Employee"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#worksForObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Address">
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="undergraduateDegreeFrom">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#undergraduateDegreeFromSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Person"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#undergraduateDegreeFromObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#University"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Advertisement">
        <rdfs:subClassOf rdf:resource="#Publication"/>
    </owl:Class>
    <owl:Class rdf:ID="teacherOf">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#teacherOfSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Faculty"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#teacherOfObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Course"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Employee">
        <rdfs:subClassOf rdf:resource="#Person"/>
    </owl:Class>
    <owl:Class rdf:ID="Organism">
        <rdfs:subClassOf rdf:resource="#PhysicalObject"/>
    </owl:Class>
    <owl:Class rdf:ID="head">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#headSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#headObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Person"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="WorkshopPaper">
        <rdfs:subClassOf rdf:resource="#Article"/>
    </owl:Class>
    <owl:Class rdf:ID="SocialGroup">
        <rdfs:subClassOf rdf:resource="#Agent"/>
    </owl:Class>
    <owl:Class rdf:ID="CommercialOrganization">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Organization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Faculty">
        <rdfs:subClassOf rdf:resource="#Employee"/>
    </owl:Class>
    <owl:Class rdf:ID="author">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#authorSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Document"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#authorObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Person"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="PersonalHomepage">
        <rdfs:subClassOf rdf:resource="#HomePage"/>
    </owl:Class>
    <owl:Class rdf:ID="Artifact">
        <rdfs:subClassOf rdf:resource="#PhysicalObject"/>
    </owl:Class>
    <owl:Class rdf:ID="member">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#memberSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#SocialGroup"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#memberObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Person"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Process">
        <rdfs:subClassOf rdf:resource="#Activity"/>
    </owl:Class>
    <owl:Class rdf:ID="Assistant">
        <rdfs:subClassOf rdf:resource="#Employee"/>
    </owl:Class>
    <owl:Class rdf:ID="subOrganizationOf">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#subOrganizationOfSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#subOrganizationOfObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom>
                    <owl:Class rdf:about="#Organization"/>
                </owl:allValuesFrom>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="EducationOrganization">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Organization"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="DoctoralThesis">
        <rdfs:subClassOf>
            <owl:Class rdf:about="#Thesis"/>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Form">
        <rdfs:subClassOf rdf:resource="#Document"/>
    </owl:Class>
    <owl:Class rdf:ID="VisitingProfessor">
        <rdfs:subClassOf rdf:resource="#Professor"/>
    </owl:Class>
    <owl:Class rdf:ID="Organization">
        <rdfs:subClassOf rdf:resource="#SocialGroup"/>
        <rdfs:subClassOf rdf:resource="#Individual"/>
    </owl:Class>
    <owl:Class rdf:ID="softwareDocumentation">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#softwareDocumentationSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Software"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#softwareDocumentationObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Publication"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Chair">
        <rdfs:subClassOf rdf:resource="#AdministrativeStaff"/>
        <rdfs:subClassOf rdf:resource="#Professor"/>
    </owl:Class>
    <owl:Class rdf:ID="Newspaper">
        <rdfs:subClassOf rdf:resource="#Periodical"/>
    </owl:Class>
    <owl:Class rdf:ID="ClericalStaff">
        <rdfs:subClassOf rdf:resource="#AdministrativeStaff"/>
    </owl:Class>
    <owl:Class rdf:ID="Promotion">
        <rdfs:subClassOf rdf:resource="#Document"/>
    </owl:Class>
    <owl:Class rdf:ID="GovernmentOrganization">
        <rdfs:subClassOf rdf:resource="#Organization"/>
    </owl:Class>
    <owl:Class rdf:ID="homeAddress">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#homeAddressSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Document"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#homeAddressObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Organization"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="SystemsStaff">
        <rdfs:subClassOf rdf:resource="#AdministrativeStaff"/>
    </owl:Class>
    <owl:Class rdf:ID="Thesis">
        <rdfs:subClassOf rdf:resource="#Publication"/>
    </owl:Class>
    <owl:Class rdf:ID="takesCourse">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#takesCourseSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Student"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
```

```
                <owl:onProperty rdf:resource="#takesCourseObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Course"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Book">
        <rdfs:subClassOf rdf:resource="#Publication"/>
    </owl:Class>
    <owl:Class rdf:ID="doctoralDegreeFrom">
        <rdfs:subClassOf rdf:resource="#Association"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#doctoralDegreeFromSubject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#Person"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
rdf:resource="#doctoralDegreeFromObject"
rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty"/>
                <owl:allValuesFrom rdf:resource="#University"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="UndergraduateStudent">
        <rdfs:subClassOf rdf:resource="#Student"/>
    </owl:Class>
    <owl:DatatypeProperty rdf:ID="orgPhone">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Organization"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="tenured">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Professor"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="publishDate"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Document"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="addressCity"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Address"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="eventStart"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Event"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="addressState"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
```

```
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Address"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="workPhone">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Person"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="value"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Lexical"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="homePhone">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Person"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="addressZip"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Address"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="softwareVersion">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Software"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="addressStreet"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Address"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="title"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Document"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="volume"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Periodical"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="emailAddress">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Person"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="eventEnd"
rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
        <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <rdfs:domain rdf:resource="#Event"/>
    </owl:DatatypeProperty>
</rdf:RDF>
```

Anexo B – Código em *OWL* das Ontologias do Segundo Estudo de Caso

Os códigos-fonte das ontologias de entrada do CATO deste estudo de caso e os de sua ontologia de saída, dos módulos sem e com ordenação alfabética, podem ser obtidos nos seguintes endereços eletrônicos:

Primeira Ontologia de entrada

http://www.inf.puc-rio.br/~cfelicissimo/person_WordNet_FirstOnto.owl

Segunda Ontologia de entrada

http://www.inf.puc-rio.br/~cfelicissimo/CMUSecondOnto.owl

Ontologia de saída do módulo sem ordenação alfabética

http://www.inf.puc-rio.br/~cfelicissimo/scdSolCombPropFinalFile.owl

Ontologia de saída do módulo com ordenação alfabética

http://www.inf.puc-rio.br/~cfelicissimo/scdSolCombPropFinalFileOd.owl

Anexo C – Código em *OWL* das Ontologias do Terceiro Estudo de Caso

Os códigos-fonte das ontologias de entrada do CATO deste estudo de caso e os de sua ontologia de saída, dos módulos sem e com ordenação alfabética, podem ser obtidos nos seguintes endereços eletrônicos:

Primeira Ontologia de entrada

http://www.inf.puc-rio.br/~cfelicissimo/vehicle_SUMO_firstOnto.owl

Segunda Ontologia de entrada

http://www.inf.puc-rio.br/~cfelicissimo/vehicle_CYC_secondOnto.owl

Ontologia de saída do módulo sem ordenação alfabética

http://www.inf.puc-rio.br/~cfelicissimo/thdSolCombPropFinalFile.owl

Ontologia de saída do módulo com ordenação alfabética

http://www.inf.puc-rio.br/~cfelicissimo/thdSolCombPropFinalFileOd.owl

Anexo D – Informações dos Métodos do CATO

Os métodos implementados no CATO são apresentado a seguir. Os métodos específicos da implementação utilizada do algoritmo *TreeDiff* podem ser encontrados em (Bergmann, 2002).

1. Método "leOnto". É encontrado nas classes nomeadas "SolCombSinonimos", "SolCombSinonimosWithOrderNodes" e "LastStepSolCombSinonimos".
    - **Descrição**: responsável pela criação de um modelo orientado a objetos do tipo *OWL* a partir de uma ontologia escrita nesta linguagem.
    - **Parâmetro de entrada**:
        - *String ontoFileName*: nome do arquivo da ontologia escrita na linguagem *OWL*.
    - **Valor retornado**:
        - *OntModel ontModelObj:* modelo da ontologia
2. Método "comparacaoSintaticaEUsoDeSinonimos". É encontrado nas classes nomeadas "SolCombSinonimos" e "SolCombSinonimosWithOrderNodes".
    - **Descrição**: responsável pela comparação lexical de cada conceito e de seu respectivo sinônimo de uma dada ontologia com cada conceito de uma outra dada ontologia. Em um primeiro momento, compara-se os nomes de conceitos; em um segundo momento, compara-se seus sinônimos, caso estes estejam cadastrados no banco de dados utilizado.
    - **Parâmetros de entrada**:
        - *OntModel ontModelObjFirstOntology*: modelo da primeira ontologia.
        - *OntModel ontModelObjSecondOntology*: modelo da segunda ontologia.
        - *String nameSpaceOnt1*: *namespace* da primeira ontologia.
        - *String nameSpaceOnt2*: *namespace* da segunda ontologia.

- **Valor retornado**:
    - *OntModel ontModelObjAux*: modelo da ontologia não analisada acrescentado com as ligações para os conceitos equivalentes da ontologia analisada.

3. Método "hashTableForSuperClassInformationEquivalentClassReplaced". É encontrado nas classes nomeadas "SolCombSinonimos", "SolCombSinonimosWithOrderNodes" e "LastStepSolCombSinonimos".
    - **Descrição**: responsável pela inserção de conceitos de uma dada ontologia em uma estrutura de dados auxiliar. Substitui, caso haja, os nomes de conceitos equivalentes por seus sinônimos para facilitar a comparação hierárquica. Caso os conceitos equivalentes encontram-se nas duas ontologias, após a substituição de um dos conceitos equivalente por seu sinônimo, os conceitos equivalentes são removidos da estrutura de dados auxiliar utilizada.
    - **Parâmetros de entrada**:
        - *OntModel ontModelObj*: modelo da ontologia.
        - *String nameSpaceOnt: namespace* da ontologia.
    - **Valor retornado**:
        - *hashTableForSuperClassInformation:* estrutura de dados do tipo tabela *Hash* que armazena conceitos e seus respectivos conceitos pais (um nível hierárquico acima) nos seus campos chaves e conteúdos, respectivamente.

4. Método "equivalentClassInformation". É encontrado nas classes nomeadas "SolCombSinonimos", "SolCombSinonimosWithOrderNodes" e "LastStepSolCombSinonimos".
    - **Descrição**: responsável pela inserção de conceitos de uma ontologia e de seus conceitos equivalentes, encontrados em uma outra ontologia, em uma estrutura de dados auxiliar.
    - **Parâmetro de entrada**:
        - *OntModel ontModelObj*: modelo da ontologia.
    - **Valor retornado**:
        - *HashTable hashTableForEquivalentClassInformation:* estrutura de dados do tipo tabela *Hash* que armazena

conceitos e seus conceitos equivalentes nos seus campos chaves e conteúdos, respectivamente.

5. Método "buildTagsXML". É encontrado nas classes nomeadas "SolCombSinonimos" e "SolCombSinonimosWithOrderNodes".
   - **Descrição**: responsável pela representação da estrutura hierárquica de uma dada ontologia em arquivo do tipo *XML*.
   - **Parâmetros de entrada**:
     - *String ontoFileName*: nome do arquivo da ontologia escrita na linguagem *OWL*.
     - *String nameSpaceOnt*: *namespace* da ontologia.
   - **Valor retornado**:
     - *String result:* variável auxiliar que armazena em *tags XML* a estrutura hierárquica de uma ontologia analisada.
6. Método "buildSubClassTags". É encontrado nas classes nomeadas "SolCombSinonimos" e "SolCombSinonimosWithOrderNodes".
   - **Descrição**: responsável pela representação da estrutura hierárquica dos sub-conceitos de um conceito analisado de uma ontologia. A criação desta estrutura hierárquica é recursiva, ou seja, continua até quando nenhum sub-conceito é mais encontrado.
   - **Parâmetros de entrada**:
     - *String className:* nome do conceito a ter seus sub-conceitos armazenados em *tags XML*.
     - *String bufferString:* variável auxiliar que armazena as *tags XML* já criadas.
     - *String ontoFileName:* nome do arquivo da ontologia escrita na linguagem *OWL*.
     - *String nameSpaceOnt: namespace* da ontologia.
   - **Valor retornado**:
     - *String result:* variável auxiliar que armazena em *tags XML* a estrutura hierárquica da ontologia analisada.
7. Método "mostraInfoDaOnto". É encontrado nas classes nomeadas "SolCombSinonimos", "SolCombSinonimosWithOrderNodes" e "LastStepSolCombSinonimos".

- **Descrição**: responsável pela exibição de cada uma das seguintes informações de uma dada ontologia: *URI* de seu conceito, *URI* de seu conceito pai (um nível hierárquico acima), *URI* de seu conceito avô (dois níveis hierárquicos acima), instâncias, *URI* do conceito equivalente, *URI* do conceito pai do conceito equivalente, *URI* do avô do conceito equivalente.
- **Parâmetro de entrada**:
    - *OntModel ontModelObj:* modelo de uma dada ontologia.
- **Valor retornado**:
    - Informação na tela.

8. Método "compare". É encontrado na classe nomeada "trace.view.DOMComparatorViewWithoutInterface".
    - **Descrição**: responsável pela comparação estrutural das hierarquias de duas árvores dadas.
    - **Parâmetro de entrada**:
    - **Valor retornado**:
        - Informação na tela.
9. Método "showSimilarities". É encontrado na classe nomeada "trace.view.DOMComparatorViewWithoutInterface".
    - **Descrição**: responsável pela exibição dos conceitos identificados como *similares*.
    - **Parâmetro de entrada**:
    - **Valor retornado**:
        - Informação na tela.
10. Método "LastUpdateEquivalentClass". É encontrado nas classes nomeadas "SolCombSinonimos", "SolCombSinonimosWithOrderNodes" e "LastStepSolCombSinonimos".
    - **Descrição**: responsável pela inserção de novos conceitos equivalentes nos arquivos "congelados" da primeira etapa da estratégia e pela geração da ontologia de saída, resultado do CATO.
    - **Parâmetros de entrada**:
        - *OntModel ontModelObjVar1*: modelo da primeira ontologia.
        - *OntModel ontModelObjVar2*: modelo da segunda ontologia.

  - *String nameSpaceFirstOnt*: *namespace* da primeira ontologia.
  - *String nameSpaceSecondOnt*: *namespace* da segunda ontologia.
- **Valor retornado**:
  - *OntModel ontModelObjVar2.union(ontModelObjVar1)*: união dos modelos das duas ontologias.

11. Método “open”. É encontrado na classe nomeada “BDQuery”.
- **Descrição**: responsável pela conexão com o banco de dados utilizado.
- **Parâmetros de entrada**:
- **Valor retornado**:
  - *Booleano true* ou *false*: retorna verdadeiro se a conexão foi aberta com o banco de dados com sucesso e falso, caso contrário.

12. Método “close”. É encontrado na classe nomeada “BDQuery”.
- **Descrição**: responsável por fechar a conexão com o banco de dados.
- **Parâmetro de entrada**:
- **Valor retornado**:

13. Método “show”. É encontrado na classe nomeada “BDQuery”.
- **Descrição**: responsável pela identificação dos sinônimos cadastrados no banco de dados utilizado.
- **Parâmetros de entrada**:
  - *String tabela*: nome da tabela do banco de dados a ser investigada.
  - *String result*: nome da coluna a ser investigada.
  - *String conceito*: nome do valor da coluna a ser investigado.
- **Valor retornado**:
  - *Vetor vetor*: vetor com todos os sinônimos encontrados cadastrados no banco de dados do parâmetro *conceito*.

Anexo E – Código em *Java* da Implementação do CATO

Os códigos-fonte do CATO são disponibilizados a seguir. Os códigos-fonte específicos da implementação utilizada do algoritmo *TreeDiff* podem ser encontrados em (Bergmann, 2002).

Para utilização do componente CATO e replicação dos resultados apresentados nesta dissertação, deve-se criar um projeto em Java com as classes descritas abaixo e executar o método "*main*" das classes SolCombSinonimos.java ou SolCombSinonimosWithOrderNodes.java. Os códigos em OWL das ontologias a serem alinhadas devem ser copiados para os arquivos "firstOnto.owl" e "secondOnto.owl", encontrados no mesmo diretório do projeto do CATO.

SolCombSinonimos.java e SolCombSinonimosWithOrderNodes.java

Estas duas classes só diferem pelo método nomeado "builtTagsXML". Na segunda classe há a função de ordenação chamada *sort* da linguagem Java que não existe na primeira classe. Por isso, apenas o código-fonte da segunda classe é disponibilizado no documento. O código-fonte da primeira classe é idêntico a este salvo a exclusão da função *sort* do método "builtTagsXML".

```
package cato;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Collections;
import java.util.Enumeration;
import java.util.Hashtable;
import java.util.Iterator;
import java.util.Vector;
import com.hp.hpl.jena.ontology.OntClass;
import com.hp.hpl.jena.ontology.OntModel;
import com.hp.hpl.jena.ontology.OntModelSpec;
import com.hp.hpl.jena.ontology.impl.OntClassImpl;
import com.hp.hpl.jena.rdf.model.Model;
import com.hp.hpl.jena.rdf.model.ModelFactory;
```

```
public class SolCombSinonimosVDefesaMestrado {

      private static String sourceDir = "file:";

      private static OntModel ontModelObjAux =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);
      private static OntModel ontModelObjAuxFirst =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);
      private static OntModel ontModelObjAuxSecond =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);

      //ESTRUTURA DE DADOS AUXILIAR PARA ARMAZENAR AS CLASSES
EQUIVALENTES DAS DUAS ONTOLOGIAS.
      private static Hashtable
hashTableForEquivalentClassInformation = new Hashtable();


      public static OntModel leOnto(String ontoFileName) {
          OntModel ontModelObj =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);

          ontModelObj.getDocumentManager().addAltEntry(sourceDir +
ontoFileName, sourceDir + ontoFileName);
            ontModelObj.read(sourceDir + ontoFileName);

            return ontModelObj;
      }


      public static OntModel
comparacaoSintaticaEUsoDeSinonimos(OntModel
ontModelObjFirstOntology, OntModel ontModelObjSecondOntology,
String nameSpaceOnt1, String nameSpaceOnt2)
      {
            if (BDQuery.open() == false) {
                  System.out.println("Erro abrindo conexao");
            } else {
                  //CONSEGUIU ABRIR A CONEXAO COM O BANCO DE
DADOS.
                  //***INICIO: INICIALIZACAO DAS VARIAVEIS
UTILIZADAS NA FUNCAO:***
                  OntClass aClass = null;
                  String nameSpaceObj = null;
                  String nameSpaceObjSynonym = null;

                  Hashtable hashTableForSecondOntologyNames = new
Hashtable();
                  Hashtable
hashTableForSecondOntologySuperClassNames = new Hashtable();
                  Hashtable
hashTableForSecondOntologySuperSuperClassNames = new Hashtable();
                  Hashtable hashTableForSecondOntologyInstances =
new Hashtable();

                  boolean allConditionsForEquivalentClasses =
false;

                  Iterator iteratorB =
ontModelObjSecondOntology.listClasses();
                  while ( iteratorB.hasNext() )
                  {
                        OntClassImpl ontObjSecondOntology =
(OntClassImpl) iteratorB.next();
                        if ((ontObjSecondOntology.getNameSpace()
!= null)
                              &&
(ontObjSecondOntology.getNameSpace()).equals(nameSpaceOnt2))
                        {
                              if
(ontObjSecondOntology.getLocalName() != null) {
                                    String
ontObjSecondOntologyName =
ontObjSecondOntology.getLocalName().trim();

      hashTableForSecondOntologyNames.put(ontObjSecondOntologyName
, ontObjSecondOntologyName);
                                    if
(ontObjSecondOntology.getSuperClass() != null)
                                    {
                                          String
ontObjSecondOntologySuperClassName =
ontObjSecondOntology.getSuperClass().getLocalName();
                                          if
(ontObjSecondOntologySuperClassName != null)
                                          {

      ontObjSecondOntologySuperClassName =
ontObjSecondOntologySuperClassName.trim();

      hashTableForSecondOntologySuperClassNames.put(ontObjSecondOn
tologyName,ontObjSecondOntologySuperClassName);
                                                if
(ontObjSecondOntology.getSuperClass().getSuperClass()!= null)
                                                {
                                                      String
ontObjSecondOntologySuperSuperClassName =
ontObjSecondOntology.getSuperClass().getSuperClass().getLocalName(
);
                                                      if
(ontObjSecondOntologySuperSuperClassName!= null)
                                                      {

      ontObjSecondOntologySuperSuperClassName =
ontObjSecondOntologySuperSuperClassName.trim();

      hashTableForSecondOntologySuperSuperClassNames.put(ontObjSec
ondOntologyName, ontObjSecondOntologySuperSuperClassName);
                                                      } else {

      hashTableForSecondOntologySuperSuperClassNames.put(ontObjSec
ondOntologyName,"***semAvo***");
                                                      }
                                                }
                                          }
                                    } else {

      hashTableForSecondOntologySuperClassNames.put(ontObjSecondOn
tologyName, "***semPai***");

      hashTableForSecondOntologySuperSuperClassNames.put(ontObjSec
ondOntologyName, "***semAvo***");
                                    }
                                    if
(ontObjSecondOntology.listInstances() != null)
                                    {
                                          for (Iterator iteratorI
= ontObjSecondOntology.listInstances(); iteratorI.hasNext();)
                                          {
                                                Object
secondOntologyInstance = (Object) iteratorI.next();
                                                int aux =
secondOntologyInstance.toString().indexOf("#");
                                                if (aux >= 0) {
                                                      String
secondOntologyInstanceName =
(secondOntologyInstance.toString().substring(aux)).trim();

      hashTableForSecondOntologyInstances.put(secondOntologyInstan
ceName, ontObjSecondOntologyName);
                                                }
                                          }
                                    }
                              }
                        }
                  }
                  Iterator iteratorA =
ontModelObjFirstOntology.listClasses();
                  while (iteratorA.hasNext()) {
                        OntClassImpl ontObjFirstOntology =
(OntClassImpl) iteratorA.next();
                        if ( (ontObjFirstOntology.getNameSpace()
!= null) &&
(ontObjFirstOntology.getNameSpace()).equals(nameSpaceOnt1) )
                        {
                              if
(ontObjFirstOntology.getLocalName() != null)
                              {
                                    String ontObjFirstOntologyName
= ontObjFirstOntology.getLocalName();
                                    ontObjFirstOntologyName =
ontObjFirstOntologyName.trim();
                                    if
(hashTableForSecondOntologyNames.get(ontObjFirstOntologyName) !=
null)
                                    {
                                          nameSpaceObj =
ontObjFirstOntology.getNameSpace();
                                          if ( (nameSpaceObj !=
null) && (nameSpaceObj.equals(nameSpaceOnt1)) )
                                          {

      nameSpaceObjSynonym = nameSpaceOnt2;
                                          } else {

      nameSpaceObjSynonym = nameSpaceOnt1;
                                          }
                                          if (
(ontObjFirstOntology.getSuperClass() != null) &&
(hashTableForSecondOntologySuperClassNames.get(ontObjFirstOntology
Name) != "***semPai***") )
                                          {
                                                String
ontObjFirstOntologySuperClassName =
ontObjFirstOntology.getSuperClass().getLocalName();
                                                if
(ontObjFirstOntologySuperClassName != null)
                                                {

      ontObjFirstOntologySuperClassName =
ontObjFirstOntologySuperClassName.trim();
                                                      if (
(ontObjFirstOntology.getSuperClass().getSuperClass() != null) &&
(hashTableForSecondOntologySuperSuperClassNames.get(ontObjFirstOnt
ologyName) != "***semAvo***") )
                                                      {
                                                            String
ontObjFirstOntologySuperSuperClassName =
ontObjFirstOntology.getSuperClass().getSuperClass().getLocalName()
;
                                                            if
(ontObjFirstOntologySuperSuperClassName != null)
                                                            {

      ontObjFirstOntologySuperSuperClassName =
ontObjFirstOntologySuperSuperClassName.trim();

      if (
(hashTableForSecondOntologySuperClassNames.get(ontObjFirstOntology
Name) != null) &&
(hashTableForSecondOntologySuperClassNames.get(ontObjFirstOntology
Name).equals(ontObjFirstOntologySuperClassName)) )

      {

      if (
(hashTableForSecondOntologySuperSuperClassNames.get(ontObjFirstOnt
ologyName) != null) &&
(hashTableForSecondOntologySuperSuperClassNames.get(ontObjFirstOnt
ologyName).equals(ontObjFirstOntologySuperSuperClassName)) )

      {

            for (Iterator iteratorI =
ontObjFirstOntology.listInstances();iteratorI.hasNext();)

            {

                  Object firstOntologyInstance = (Object)
iteratorI.next();

                  int aux =
firstOntologyInstance.toString().indexOf("#");

                  if (aux >= 0) {

                        String firstOntologyInstanceName =
(firstOntologyInstance.toString().substring(aux)).trim();

                        if
(hashTableForSecondOntologyInstances.get(firstOntologyInstanceName
) == null)
```

```
                        {
                              allConditionsForEquivalentClasses =
false;
                        } else {
                              allConditionsForEquivalentClasses =
true;
                        }
                  }
                  if (allConditionsForEquivalentClasses) {
                        aClass =
ontModelObjSecondOntology.createClass(nameSpaceObjSynonym +
ontObjFirstOntologyName);

      aClass.addEquivalentClass(ontObjFirstOntology);
                  }
            }
            if (allConditionsForEquivalentClasses) {
                  aClass =
ontModelObjSecondOntology.createClass(nameSpaceObjSynonym +
ontObjFirstOntologyName);
                  aClass.addEquivalentClass(ontObjFirstOntology);
            }
      }
      }
                                                            }
                                                      }
                                                }
                                          }
                                    }
                                    Vector result =
BDQuery.show("Sinonimo", "tblSinonimos", ontObjFirstOntologyName);
                                    try {
                                          if (!result.isEmpty()) {

      System.out.println("Sinonimos de " +
ontObjFirstOntology.getURI() + ": " + result);
                                                Iterator syn =
result.iterator();
                                                while
(syn.hasNext()) {
                                                      String
sinonimoTermo = (String) syn.next();
                                                      if
(sinonimoTermo != null) {
      sinonimoTermo = sinonimoTermo.trim();
                                                            if
(hashTableForSecondOntologyNames.get(sinonimoTermo) != null)
                                                            {

      nameSpaceObj = ontObjFirstOntology.getNameSpace();

      if ( (nameSpaceObj != null) &&
(nameSpaceObj.equals(nameSpaceOnt1)) )

      {

      nameSpaceObjSynonym = nameSpaceOnt2;

      } else {

      nameSpaceObjSynonym = nameSpaceOnt1;

      }

      if ( (ontObjFirstOntology.getSuperClass() != null) &&
(hashTableForSecondOntologySuperClassNames.get(sinonimoTermo) !=
"***semPai***") )

      {

      String ontObjFirstOntologySuperClassName =
ontObjFirstOntology.getSuperClass().getLocalName();

      if (ontObjFirstOntologySuperClassName != null) {

            ontObjFirstOntologySuperClassName =
ontObjFirstOntologySuperClassName.trim();

            if (
(ontObjFirstOntology.getSuperClass().getSuperClass() != null ) &&
(hashTableForSecondOntologySuperSuperClassNames.get(sinonimoTermo)
!= "***semAvo***") )

            {

                  String ontObjFirstOntologySuperSuperClassName =
ontObjFirstOntology.getSuperClass().getSuperClass().getLocalName()
;

                  if (ontObjFirstOntologySuperSuperClassName !=
null) {

                        ontObjFirstOntologySuperSuperClassName =
ontObjFirstOntologySuperSuperClassName.trim();

                        if (
(hashTableForSecondOntologySuperClassNames.get(sinonimoTermo) !=
null) &&
(hashTableForSecondOntologySuperClassNames.get(sinonimoTermo).equa
ls(ontObjFirstOntologySuperClassName)) )

                        {

                              if (
(hashTableForSecondOntologySuperSuperClassNames.get(sinonimoTermo)
!= null) &&
(hashTableForSecondOntologySuperSuperClassNames.get(sinonimoTermo)
.equals(ontObjFirstOntologySuperSuperClassName)) )

                              {

                                    for (Iterator iteratorI =
ontObjFirstOntology.listInstances(); iteratorI.hasNext();)

                                    {

                                          Object
firstOntologyInstance = (Object) iteratorI.next();

                                          int aux =
firstOntologyInstance.toString().indexOf("#");

                                          if (aux >= 0) {

                                                String
firstOntologyInstanceName =
firstOntologyInstance.toString().substring(aux);

                                                if
(hashTableForSecondOntologyInstances.get(firstOntologyInstanceName
) == null)

                                                {


      allConditionsForEquivalentClasses = false;

                                                } else {


      allConditionsForEquivalentClasses = true;

                                                }

                                          }

                                          if
(allConditionsForEquivalentClasses) {

                                                aClass =
ontModelObjSecondOntology.createClass(nameSpaceObjSynonym +
sinonimoTermo);


      aClass.addEquivalentClass(ontObjFirstOntology);

                                          }

                                    }

                                    if
(allConditionsForEquivalentClasses) {

                                          aClass =
ontModelObjSecondOntology.createClass(nameSpaceObjSynonym +
sinonimoTermo);


      aClass.addEquivalentClass(ontObjFirstOntology);

                                    }

                              }

                        }

                  }

            }

      }

      }
                                                      }
                                                }
                                          }
                                    }
                                    } catch (Exception e) {

      System.out.println(e.toString());
                                          e.printStackTrace();
                                    }
                              }
                        }
                  }
                  BDQuery.close();
                  ontModelObjAux = (OntModel)
ontModelObjSecondOntology;
            }
            return ontModelObjAux;
      }


      public static Hashtable
hashTableForSuperClassInformationEquivalentClassReplaced(OntModel
ontModelObj, String nameSpaceOnt)
      {
            Hashtable hashTableForSuperClassInformation = new
Hashtable();

            for (Iterator iteratorA =
ontModelObj.listClasses();iteratorA.hasNext();) {
                  OntClassImpl ont = (OntClassImpl)
iteratorA.next();

                  if ( (ont.getNameSpace() != null) &&
(ont.getNameSpace()).equals(nameSpaceOnt)) {
                        if (ont.getLocalName() != null) {
                              String classNameAndURI = (String)
ont.getURI().trim();
                              String className = (String)
ont.getLocalName().trim();
                              if (ont.getSuperClass() != null) {
                                    if
(ont.getSuperClass().getLocalName() != null) {
```

```
                                          String
superClassNameAndURI = (String)
ont.getSuperClass().getURI().trim();
                                          String superClassName =
(String) ont.getSuperClass().getLocalName().trim();
                                          if
(hashTableForEquivalentClassInformation.get(classNameAndURI) ==
null) {
                                                if
(hashTableForEquivalentClassInformation.get(superClassNameAndURI)
!= null) {
                                                      int
auxSuperClassName =
hashTableForEquivalentClassInformation.get(superClassNameAndURI).t
oString().indexOf("#") + 1;
                                                      if
(auxSuperClassName >= 0) {
                                                            String
contentSuperClassName =
(hashTableForEquivalentClassInformation.get(superClassNameAndURI))
.toString().substring(auxSuperClassName);

      contentSuperClassName.trim();

      hashTableForSuperClassInformation.put(className,
contentSuperClassName);
                                                      }
                                                } else {
                                                      //NAO TEM A
INFORMACAO DE CLASSE EQUIVALENTE PARA A SUPER CLASSE

      hashTableForSuperClassInformation.put(className,
superClassName);
                                                }
                                          } else {
                                                //TEM A INFORMACAO
DE CLASSE EQUIVALENTE
                                                if (
hashTableForEquivalentClassInformation.get(hashTableForEquivalentC
lassInformation.get(classNameAndURI)) != null )
                                                {
                                                      //TEM A
INFORMACAO DE CLASSE EQUIVALENTE NAS DUAS ONTOLOGIAS
                                                      //Removendo
as informacaos da URI e pegando apenas o nome "cru" da Classe .
                                                      int
auxClassName =
hashTableForEquivalentClassInformation.get(classNameAndURI).toStri
ng().indexOf("#") + 1;
                                                      if
(auxClassName >= 0) {
                                                            String
contentName
=(hashTableForEquivalentClassInformation.get(classNameAndURI)).toS
tring().substring(auxClassName);

      contentName.trim();


      //SUBSTITUI O NOME DA CLASSE PELO NOME DA SUA CLASSE
EQUIVALENTE
```

```
      hashTableForSuperClassInformation.put(contentName,
superClassName);
                                                       }

      hashTableForEquivalentClassInformation.remove(hashTableForEq
uivalentClassInformation.get(classNameAndURI));

      //hashTableForEquivalentClassInformation.remove(classeAndURI
);
                                                 } else {
                                                       //TEM A
INFORMACAO DE CLASSE EQUIVALENTE EM UMA SO ONTOLOGIA
                                                       if
(hashTableForEquivalentClassInformation.get(superClassNameAndURI)
!= null)
                                                       {
                                                             //TEM
A INFORMACAO DE CLASSE EQUIVALENTE PARA A SUPER CLASSE

                                                             int
auxSuperClassName =
hashTableForEquivalentClassInformation.get(superClassNameAndURI).t
oString().indexOf("#") + 1;
                                                             if
(auxSuperClassName >= 0) {

      String contentSuperClassName =
(hashTableForEquivalentClassInformation.get(superClassNameAndURI))
.toString().substring(auxSuperClassName);

      contentSuperClassName.trim();


      hashTableForSuperClassInformation.put(className,
contentSuperClassName);
                                                             }
                                                       } else {
                                                             //TEM
A INFORMACAO DE CLASSE EQUIVALENTE EM UMA SO ONTOLOGIA E NAO TEM A
INFORMACAO DE CLASSE EQUIVALENTE PARA A SUPER CLASSE
                                                             int
auxClassName =
hashTableForEquivalentClassInformation.get(classNameAndURI).toStri
ng().indexOf("#") + 1;
                                                             if
(auxClassName >= 0) {

      String contentName =
(hashTableForEquivalentClassInformation.get(classNameAndURI)).toSt
ring().substring(auxClassName);

      contentName.trim();


      //SUBSTITUI O NOME DA CLASSE PELO NOME DA SUA CLASSE
EQUIVALENTE

      hashTableForSuperClassInformation.put(contentName,
superClassName);
                                                             }
```

```
                                                      }
                                                }
                                          }
                                          //NAO TEM A INFORMACAO
DO NOME DA SUPERCLASSE
                                    } else if (

      hashTableForEquivalentClassInformation.get(classNameAndURI)
== null) {

      hashTableForSuperClassInformation.put(className,
"***semPai***");
                                          //TEM A INFORMACAO DE
CLASSE EQUIVALENTE
                                    } else {
                                          int auxClassName =
(hashTableForEquivalentClassInformation.get(classNameAndURI)).toSt
ring().indexOf("#") + 1;
                                          if (auxClassName >= 0) {
                                                String contentName
=
(hashTableForEquivalentClassInformation.get(classNameAndURI)).toSt
ring().substring(auxClassName);

      contentName.trim();

                                                //SUBSTITUI O NOME
DA CLASSE PELO NOME DA SUA CLASSE EQUIVALENTE

      hashTableForSuperClassInformation.put(contentName,
"***semPai***");
                                          }
                                    }
                              } else if (

      hashTableForEquivalentClassInformation.get(className) ==
null) {

      hashTableForSuperClassInformation.put(className,
"***semPai***");
                              } else {
                                    int auxClassName =
(hashTableForEquivalentClassInformation.get(classNameAndURI)).toSt
ring().indexOf("#") + 1;
                                    if (auxClassName >= 0) {
                                          String contentName =
(hashTableForEquivalentClassInformation.get(classNameAndURI)).toSt
ring().substring(auxClassName);
                                          contentName.trim();

                                          //SUBSTITUI O NOME DA
CLASSE PELO NOME DA SUA CLASSE EQUIVALENTE

      hashTableForSuperClassInformation.put(contentName,
"***semPai***");
                                    }
                              }
                        }
                  }
            }
            return hashTableForSuperClassInformation;
      }

      public static Hashtable equivalentClassInformation(OntModel
ontModelObj) {
            Hashtable hashTableForEquivalentClassInformation = new
Hashtable();

            String classNameAndURI = new String();
            String equivalentClassNameAndURI = new String();

            for (Iterator iteratorA =
ontModelObj.listClasses();iteratorA.hasNext();) {
                  OntClassImpl ont = (OntClassImpl)
iteratorA.next();

                  if (ont.getLocalName() != null) {
                        classNameAndURI = ont.getURI();
                        classNameAndURI.trim();
                  }

                  if (ont.getEquivalentClass() != null) {
                        equivalentClassNameAndURI =
ont.getEquivalentClass().getURI();
                        equivalentClassNameAndURI.trim();

      hashTableForEquivalentClassInformation.put(classNameAndURI,
equivalentClassNameAndURI);
                  }
            }
            return hashTableForEquivalentClassInformation;
      }

      public static String buildTagsXML(String ontoFileName,
String nameSpaceOnt) {
            String result = new String();
            Vector vetorOrdenadoSubClasses = new Vector();
            Vector vetorOrdenadoSubClassesNEW = new Vector();
            boolean classeImportada = false;

            Hashtable showSecondOntoSuperClassInformation =
hashTableForSuperClassInformationEquivalentClassReplaced(leOnto(on
toFileName), nameSpaceOnt);

            Enumeration contentSecondOnto =
showSecondOntoSuperClassInformation.elements();
            Enumeration keysSecondOnto =
showSecondOntoSuperClassInformation.keys();

            System.out.println("");
            System.out.println("");
            System.out.println("***CLASSES E SUPER CLASSES COM UM
DOS NOMES DE CLASSE EQUIVALENTE SUBSTITUIDOS:***");

            while (contentSecondOnto.hasMoreElements() &&
keysSecondOnto.hasMoreElements()) {

                String contentSecondOntoValue = (String)
contentSecondOnto.nextElement();
                  String keySecondOntoValue = (String)
keysSecondOnto.nextElement();
```

```
                  System.out.println("Class: " +
keySecondOntoValue + "   SuperClass: " + contentSecondOntoValue);

                  if
(contentSecondOntoValue.equals("***semPai***")) {

      vetorOrdenadoSubClasses.add(keySecondOntoValue);
                  } else if (
!(contentSecondOntoValue.equals("***semPai***")) &&
(showSecondOntoSuperClassInformation.get(contentSecondOntoValue)
== null) )
                  {

      vetorOrdenadoSubClassesNEW.add(keySecondOntoValue);
                        classeImportada = true;
                  }
            }

            if (classeImportada) {
                  Collections.sort(vetorOrdenadoSubClassesNEW);

                  for (Enumeration e =
vetorOrdenadoSubClassesNEW.elements(); e.hasMoreElements();) {
                        //Pega os valores das classes ordenados
alfabeticamente.
                        String newOrderKeyValue = (String)
e.nextElement();

                        //Precisa novamente percorrer a tabela
para pegar o nome das classes pais importadas.
                        Hashtable
auxShowSecondOntoSuperClassInformation =
hashTableForSuperClassInformationEquivalentClassReplaced(leOnto(on
toFileName), nameSpaceOnt);

                        Enumeration auxContentSecondOnto =
auxShowSecondOntoSuperClassInformation.elements();
                        Enumeration auxKeysSecondOnto =
auxShowSecondOntoSuperClassInformation.keys();

                        while (
auxContentSecondOnto.hasMoreElements() &&
auxKeysSecondOnto.hasMoreElements() ) {
                              String contentSecondOntoValue =
(String) auxContentSecondOnto.nextElement();
                              String keySecondOntoValue = (String)
auxKeysSecondOnto.nextElement();

                              //Achou a classe que tem a
superclasse importada.  Recupera o nome da superclasse da Tabela
com informacoes de SuperClasses.
                              if (
newOrderKeyValue.equals(keySecondOntoValue) ) {
                                    result = result +
buildSubClassTags(newOrderKeyValue, "<Classe>" +
contentSecondOntoValue + "<subClasse>" + newOrderKeyValue,
ontoFileName, nameSpaceOnt) + "</subClasse>" + "</Classe>";
                              }
                        }
                  }
            } else { //Nao tem classes importadas.
```

```
                  Collections.sort(vetorOrdenadoSubClasses);

                  for (Enumeration e =
vetorOrdenadoSubClasses.elements(); e.hasMoreElements();) {
                        String newOrderKeyValue = (String)
e.nextElement();
                        result = result +
buildSubClassTags(newOrderKeyValue, "<Classe>" + newOrderKeyValue,
ontoFileName, nameSpaceOnt) + "</Classe>";
                  }
            }

            result = "<ontoInXML>" + result + "</ontoInXML>";

            System.out.println("");
            System.out.println("***ESTRUTURA HIERARQUICA DA
ONTOLOGIA EXPRESSA EM TAGS DO TIPO XML:***");
            System.out.println(result);

            return result;
      }

      public static String buildSubClassTags(String className,
String bufferString, String ontoFileName, String nameSpaceOnt) {

          Hashtable showSuperClassInformation = new Hashtable();
          Vector vetorOrdenadoSubClasses = new Vector();

            showSuperClassInformation =
hashTableForSuperClassInformationEquivalentClassReplaced(leOnto(on
toFileName), nameSpaceOnt);

            Enumeration content =
showSuperClassInformation.elements();
            Enumeration key = showSuperClassInformation.keys();

            //Com o valor da Chave, busca os conteudos
            while ( content.hasMoreElements() &&
key.hasMoreElements() ) {

                String newContentValue = (String)
content.nextElement();
                  String newKeyValue = (String) key.nextElement();

                  if (newContentValue.equals(className)) {
                        vetorOrdenadoSubClasses.add(newKeyValue);
                  }
            }
            //Ordena alfabeticamente os nos
            Collections.sort(vetorOrdenadoSubClasses);

            for (Enumeration e =
vetorOrdenadoSubClasses.elements(); e.hasMoreElements();) {

                String newOrderKeyValue = (String)
e.nextElement();

                  bufferString = bufferString + "<subClasse>" +
newOrderKeyValue;
```

```
                  bufferString = 
buildSubClassTags(newOrderKeyValue, bufferString, ontoFileName, 
nameSpaceOnt);
                  bufferString = bufferString + "</subClasse>";
            }
            return bufferString;
      }

      public static Model LastUpdateEquivalentClass(OntModel 
ontModelObjVar1, OntModel ontModelObjVar2, String 
nameSpaceFirstOnt, String nameSpaceSecondOnt)
      {
            String auxLine = "";
            String className;
            String equivalentClassName;

            try {
                  BufferedReader auxFile = new BufferedReader(new 
FileReader("treeDiffResults.txt"));

                  auxLine = auxFile.readLine();
                  while (auxLine != null) {
                        int auxLineNumber = auxLine.indexOf("-");
                        if (auxLineNumber >= 0) {
                              className = auxLine.substring(0, 
auxLineNumber);
                              equivalentClassName = 
auxLine.substring(auxLineNumber + 1);
                              System.out.println("className: " + 
className + " --- equivalentClassName: " + equivalentClassName);
                              System.out.println("");

                              //A CLASSE NAO FOI IDENTIFICADA COMO 
EQUIVALENTE NAS ETAPAS ANTERIORES
                              if ( 
(hashTableForEquivalentClassInformation.get(nameSpaceFirstOnt + 
className) == null) || 
(hashTableForEquivalentClassInformation.get(nameSpaceSecondOnt + 
className) == null) )
                              {
                                    OntClass aClass = null;

                                    //PERCORRE A PRIMEIRA 
ONTOLOGIA PARA SABER SE A CLASSE EQUIVALENTE A PERTENCE
                                    Iterator iteratorA = 
ontModelObjVar1.listClasses();
                                    while (iteratorA.hasNext()) {
                                          //PERCORRE CADA CLASSE 
DA PRIMEIRA ONTOLOGIA
                                          OntClassImpl 
ontObjFirstOntology = (OntClassImpl) iteratorA.next();
                                          if ( 
(ontObjFirstOntology.getNameSpace() != null) && 
(ontObjFirstOntology.getNameSpace()).equals(nameSpaceFirstOnt) )
                                          {
                                                //HA A NECESSIDADE 
DESSA VERIFICACAO PARA ONTOLOGIAS QUE IMPORTAM OUTRAS.

                                                if ( 
(ontObjFirstOntology.getLocalName() != null) && 
(ontObjFirstOntology.getLocalName().equals(className)) ) {
```

```
                                                      aClass =
ontModelObjVar2.createClass(nameSpaceSecondOnt +
equivalentClassName);

      aClass.addEquivalentClass(ontObjFirstOntology);
                                                }
                                          }
                                    }

                                    //PERCORRE A SEGUNDA ONTOLOGIA
PARA SABER SE A CLASSE EQUIVALENTE A PERTENCE
                                    Iterator iteratorB =
ontModelObjVar2.listClasses();
                                    while (iteratorB.hasNext()) {
                                          //PERCORRE CADA CLASSE
DA PRIMEIRA ONTOLOGIA
                                          OntClassImpl
ontObjSecondOntology = (OntClassImpl) iteratorB.next();
                                          if (
(ontObjSecondOntology.getNameSpace() != null) &&
(ontObjSecondOntology.getNameSpace()).equals(nameSpaceSecondOnt) )
                                          {
                                                if (
(ontObjSecondOntology.getLocalName() != null) &&
(ontObjSecondOntology.getLocalName().equals(className)) )
                                                {
                                                      aClass =
ontModelObjVar1.createClass(nameSpaceFirstOnt +
equivalentClassName);

      aClass.addEquivalentClass(ontObjSecondOntology);
                                                }
                                          }
                                    }
                              }
                        }
                        auxLine = auxFile.readLine();
                  }
                  auxFile.close();
            } catch (IOException e) {
                  e.printStackTrace();
            }
            return ontModelObjVar2.union(ontModelObjVar1);
      }

      public static void mostraInfoDaOnto(OntModel ontModelObj) {

            System.out.println("");
            System.out.println("");
            System.out.println("***INFORMACOES DAS CLASSES DAS
ONTOLOGIAS:***");

            for (Iterator iteratorA =
ontModelObj.listClasses();iteratorA.hasNext();) {
                  OntClassImpl ont = (OntClassImpl)
iteratorA.next();

                  if (ont.getLocalName() != null) {
                        System.out.println("");
                        System.out.println("URI da Classe:" +
ont.getURI());
```

```
                        if (ont.getSuperClass() != null) {
                              if
(ont.getSuperClass().getLocalName() != null) {
                                    System.out.println("URI da
Classe Pai: " + ont.getSuperClass().getURI());

                                    if
(ont.getSuperClass().getSuperClass() != null) {
                                          if
(ont.getSuperClass().getSuperClass().getLocalName() != null) {

      System.out.println("URI da Classe Avo: " +
ont.getSuperClass().getSuperClass().getURI());
                                          }
                                    }
                              }
                        }
                  }

                  if (ont.listInstances() != null) {
                        for (Iterator iteratorI =
ont.listInstances(); iteratorI.hasNext();) {
                              Object intances = (Object)
iteratorI.next();
                              //System.out.println( "Instancias: "
+ intances );
                              int aux =
intances.toString().indexOf("#");
                              if (aux >= 0) {
                                    String instanceName =
intances.toString().substring(aux);
                                    System.out.println("
Instancias: " + instanceName);
                              }
                        }
                  }
                  if (ont.getEquivalentClass() != null) {
                        System.out.println("URI da Classe
Equivalente: " + ont.getEquivalentClass());
                  }
            }
      }

      public static void main(String[] args) throws IOException {
            OntModel ontModelObjVar1 = leOnto("firstOnto.owl");
            OntModel ontModelObjVar2 = leOnto("secondOnto.owl");
            ontModelObjAuxFirst =
comparacaoSintaticaEUsoDeSinonimos(ontModelObjVar1,
ontModelObjVar2, "file:firstOnto.owl#", "file:secondOnto.owl#");

            File saidaFirstOnto = new File("newSecondOnto.owl");
            FileWriter outFirstOnto = new
FileWriter(saidaFirstOnto);
            ontModelObjAuxFirst.write(outFirstOnto, "RDF/XML-
ABBREV");
            ontModelObjAuxSecond =
comparacaoSintaticaEUsoDeSinonimos(ontModelObjVar2,
ontModelObjVar1, "file:secondOnto.owl#", "file:firstOnto.owl#");

            File saidaSecondOnto = new File("newFirstOnto.owl");
            FileWriter outSecondOnto = new
FileWriter(saidaSecondOnto);
            ontModelObjAuxSecond.write(outSecondOnto, "RDF/XML-
ABBREV");

            Model ontModelObjAuxFinal =
ontModelObjAuxFirst.union(ontModelObjAuxSecond);

            File saidaThirdOnto = new File("mappingOnto.owl");
            saidaThirdOnto.createNewFile();
            FileWriter out1 = new FileWriter(saidaThirdOnto);
            ontModelObjAuxFinal.write(out1, "RDF/XML-ABBREV");

            mostraInfoDaOnto(leOnto("mappingOnto.owl"));
            System.out.println("");
            System.out.println("");
            System.out.println("***FIM DA PRIMEIRA ETAPA DA
SOLUCAO COMBINADA PROPOSTA***");
            System.out.println("");
            hashTableForEquivalentClassInformation =
equivalentClassInformation(leOnto("mappingOnto.owl"));
            String firstResult = buildTagsXML("firstOnto.owl",
"file:firstOnto.owl#");

            File firstForTreeDiff = new
File("newFirstOntoForTreeDiff.xml");
            FileOutputStream newFirstOntoForTreeDiff = new
FileOutputStream(firstForTreeDiff);
            newFirstOntoForTreeDiff.write(firstResult.getBytes());
            newFirstOntoForTreeDiff.close();

            String secondResult = buildTagsXML("secondOnto.owl",
"file:secondOnto.owl#");
            File secondForTreeDiff = new
File("newSecondOntoForTreeDiff.xml");
            FileOutputStream newSecondOntoForTreeDiff =     new
FileOutputStream(secondForTreeDiff);

      newSecondOntoForTreeDiff.write(secondResult.getBytes());
            newSecondOntoForTreeDiff.close();

            System.out.println("");
            System.out.println("");
            System.out.println("***FIM DA ETAPA INTERMEDIARIA
(PRIMEIRA ETAPA -> SEGUNDA ETAPA)***");
            System.out.println("");
            System.out.println("");
      }

}
```

BDQuery.java

```
package cato;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Vector;


public class BDQuery {

      private static Connection bdAccess;

      public static boolean open() {
            // Register jdbc driver:
            try {
                  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            } catch (ClassNotFoundException e) {
                  e.printStackTrace();
                  return false;
            }

            // Connect to DB:
            try {
                  bdAccess =
DriverManager.getConnection("jdbc:odbc:CATO", "", "");
            } catch (SQLException se) {
                  se.printStackTrace();
                  return false;
            }
            return true;
      }

      public static void close() {
            try {
                  bdAccess.close();
            } catch (SQLException se) {
                  se.printStackTrace();
            }
      }

      public static Vector show(String result, String tabela,
String termo) {
            // Do the Query:
            Vector vetor = new Vector();
            try {
                  String selectSQL = "select " + result + " from "
+ tabela + " where conceito = '" + termo + "' order by " + result;
                  Statement stmt = bdAccess.createStatement();
                  ResultSet rset = stmt.executeQuery(selectSQL);
                  while (rset.next()) {
                        String resultado = rset.getString(1);
                        if (resultado != null) {
                              //Retorna resultados linha a linha:
                              //System.out.println(resultado);
                              vetor.add(resultado);
                        }
                  }
                  stmt.close();
            } catch (SQLException se) {
                  se.printStackTrace();
            }
            return vetor;
      }

//          public static void main( String[] args ) {
//                      BDQuery bdqueryobj = new BDQuery();
//                      System.out.println( "Sinonimos: " + ( 
bdqueryobj.show( "Sinonimos", "tblSinonimos", "car" ) ) );
//                      System.out.println( "Termos Relacionados: 
" + ( bdqueryobj.show( "TermosRelacionados", 
"tblTermosRelacionados", "car" ) ) );
//                }

}


/*OUTRA MANEIRA PARA CONEXAO COM O BD:
try
{
      // Step 1. Load the Type 1 database driver:
      Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
      // Step 2. Create a connection to the database
      //         using pre-defined Data Source:
      Connection con = DriverManager.getConnection
            ("jdbc:odbc:PreDefinedDB",userid,password);
      // Step 3. Create a Statement object:
      Statement stmt = con.createStatement();
      // Step 4. Build and send the database SQL query:
      String query = "SELECT * FROM DataTable";
      // Step 5. Get the resultset from the database:
      ResultSet rs = stmt.executeQuery(query);
      // Step 6. Process the data:
      // ...
}
*/
```

DOMComparatorViewWithoutInterface.java

```
package trace.view;


import java.io.*;
import java.util.Iterator;
import java.util.Vector;
import org.w3c.dom.*;
import trace.delta.dom.*;
import trace.result.Similarity;


public class DOMComparatorViewWithoutInterface  {
      DOMChangeManager changeManager = null;

      String dir = "D:/Carol/eclipse-SDK-3.0-
win32/workspace/CATO/";

      String left = dir + "newFirstOntoForTreeDiff.xml";
      String right = dir + "newSecondOntoForTreeDiff.xml";


      public void compare() {
            org.w3c.dom.Document aFirstDoc = null;
            try {
                  ulf.dom.UlfDOMParserWrapper aux = new
ulf.dom.UlfDOMParserWrapper();
                  aFirstDoc = aux.parse(left);
                  ulf.dom.DOMUtilities.normalize(aFirstDoc);
            } catch(Exception e) {
                  e.printStackTrace();
            }

            org.w3c.dom.Document aSecDoc = null;
            try {
                  ulf.dom.UlfDOMParserWrapper aux = new
ulf.dom.UlfDOMParserWrapper();
                  aSecDoc = aux.parse(right);
                  ulf.dom.DOMUtilities.normalize(aSecDoc);
            } catch(Exception e) {
                  e.printStackTrace();
            }

            FindDOMDifferenceStrategy aStrategy = new
trace.delta.dom.LargestApproximatelyCommonTraceSubTree();
            DOMComparator aComp = new DOMComparator();
            aStrategy.setComparator(aComp);
            trace.delta.dom.similarity.FindAtomicSimilarity
aSimStrategy = new
trace.delta.dom.similarity.FindAtomicSimilarity(aComp);

            changeManager = new DOMChangeManager(aFirstDoc,
aSecDoc, aStrategy, aSimStrategy);
            changeManager.findChanges();

            // Adicionado
            changeManager.computeSimilarity();

            showSimilarities();
      }


      public void showSimilarities() {
            try {
                  File saidaAuxOnto = new File(dir + 
"treeDiffResults.txt");
                  saidaAuxOnto.createNewFile();
                  BufferedWriter auxFile = new BufferedWriter(new 
FileWriter(saidaAuxOnto));

                  Vector armazenaSim = new Vector();
                  armazenaSim = changeManager.getSimilarities();

                  Node originalNodeName = null;

                  System.out.println("");
                  System.out.println("Similarities Level:");

                  for (Iterator iter = armazenaSim.iterator(); 
iter.hasNext();) {
                        Similarity sim = (Similarity) iter.next();

                        NodeList newTreeNodeList = 
sim.getNewTreeNode().getChildNodes();
                        NodeList originalTreeNodeList = 
sim.getOriginalTreeNode().getChildNodes();

                        for (int i = 0; i < 
newTreeNodeList.getLength(); i++) {
                              Node originalTreeNode = (Node) 
originalTreeNodeList.item(i);
                              Node newTreeNode = (Node) 
newTreeNodeList.item(i);

                              if ( (originalTreeNode != null) && 
(newTreeNode != null) ){

                                    if ( 
(originalTreeNode.getNodeValue() != null) && 
(newTreeNode.getNodeValue() != null) ) {

      System.out.println(originalTreeNode.getNodeValue() + " -> " 
+  newTreeNode.getNodeValue() +  "   *** Similarity Level: " + 
sim.getLevel()*100 + "%");

                                          if ( 
(sim.getLevel()*100) >= 75) {
                                                //ESCREVE A 
INFORMACAO EM UM ARQUIVO DO TIPO TEXTO PARA PASSAGEM DE INFORMACAO
                                                auxFile.write( 
(originalTreeNode.getNodeValue()) + "-" + 
(newTreeNode.getNodeValue()) );
                                                auxFile.newLine();

                                          }
                                    }
                              }
                        }
                  }
                  auxFile.close();
            }
            catch (IOException e) {
                  e.printStackTrace();
            }
      }


      public static void main(java.lang.String[] args) {
            try {
                  TreeDiff aDOMComparatorView = new TreeDiff();
                  aDOMComparatorView.compare();
                  } catch (Throwable exception) {
                        exception.printStackTrace(System.out);
                        }
            }
}
```

LastStepSolCombSinonimos.java

```
    package cato;



    import java.io.BufferedReader;
    import java.io.File;
    import java.io.FileReader;
    import java.io.FileWriter;
    import java.io.IOException;
    import java.util.Hashtable;
    import java.util.Iterator;
    import com.hp.hpl.jena.ontology.OntClass;
    import com.hp.hpl.jena.ontology.OntModel;
    import com.hp.hpl.jena.ontology.OntModelSpec;
    import com.hp.hpl.jena.ontology.impl.OntClassImpl;
    import com.hp.hpl.jena.rdf.model.Model;
    import com.hp.hpl.jena.rdf.model.ModelFactory;



    public class LastStepSolCombSinonimos {

        private static String sourceDir = "file:";

     //CRIANDO AS VARIAVEIS PARA OS MODELOS DAS ONTOLOGIAS.  A
API JENA TRANSFORMA CADA ONTOLOGIA EXPRESSA EM
     //UM ARQUIVO DO TIPO OWL, POR EXEMPLO, EM UM MODELO E
TRABALHA NELE.  PRECISA-SE DESTA TRANSFORMACAO.
     private static OntModel ontModelObjAux =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);
     private static OntModel ontModelObjAuxFirst =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);
     private static OntModel ontModelObjAuxSecond =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);

     //ESTRUTURA DE DADOS AUXILIAR PARA ARMAZENAR AS CLASSES
EQUIVALENTES DAS DUAS ONTOLOGIAS.
     private static Hashtable
hashTableForEquivalentClassInformation = new Hashtable();
```

```
    //FUNCAO QUE TRANSFORMA UM ONTOLOGIA EXPRESSA EM UM ARQUIVO
DO TIPO OWL EM UM MODELO ONTOLOGICO.
    //LE O ARQUIVO E RETORNA O SEU MODELO.
    public static OntModel leOnto(String address) {
        OntModel ontModelObj =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);

        ontModelObj.getDocumentManager().addAltEntry(address,
address);
        ontModelObj.read(sourceDir + address);

        return ontModelObj;
    }

    //FUNCAO QUE CADASTRA EM UMA TABELA HASH NO CAMPO CHAVE O
NOME DA CLASSE E NO CAMPO CONTEUDO O NOME
    //DA SUPER CLASSE.  SUBSTITUI, CASO HAJA, OS NOMES DE
CLASSES EQUIVALENTES PARA FACILITAR A COMPARACAO HIERARQUICA.
    public static Hashtable
hashTableForSuperClassInformationEquivalentClassReplaced(OntModel
ontModelObj, String nameSpaceOnt)
    {
        Hashtable hashTableForSuperClassInformation = new
Hashtable();
        String classURI = new String();
        String className = new String();
        String superClassURI = new String();
        String superClassName = new String();

        for (Iterator iteratorA =
ontModelObj.listClasses();iteratorA.hasNext();)
        {
            OntClassImpl ont = (OntClassImpl)
iteratorA.next();
            if ( (ont.getNameSpace() != null) &&
(ont.getNameSpace()).equals(nameSpaceOnt) )
            {
                //HA A NECESSIDADE DESSA VERIFICACAO PARA
ONTOLOGIAS QUE IMPORTAM OUTRAS.
                if (ont.getLocalName() != null)
                {
```

```
                              classURI         =         (String)
ont.getURI().trim();
                              className         =        (String)
ont.getLocalName().trim();

                              if  (  (ont.getSuperClass()  !=  null)
&& (ont.getSuperClass().getLocalName() != null) )
                              {
                                    for   (Iterator   iteratorB   =
ont.listSuperClasses(); iteratorB.hasNext();)
                                    {
                                        OntClassImpl    ontPai    =
(OntClassImpl) iteratorB.next();
                                          if ( (ontPai != null) &&
(ontPai.getLocalName() != null) ) {
                                              superClassURI       =
(String) ontPai.getURI().trim();
                                                superClassName    =
(String) ontPai.getLocalName().trim();

                                                if
(hashTableForEquivalentClassInformation.get(classURI) == null)
                                                {
                                                      //NAO  TEM  A
INFORMACAO DE CLASSE EQUIVALENTE
                                                      if          (
hashTableForEquivalentClassInformation.get(superClassURI)  !=  null
)
                                                      {
                                                            //TEM
A INFORMACAO DE CLASSE EQUIVALENTE PARA A SUPER CLASSE

                                                            int
auxSuperClassName                                               =
hashTableForEquivalentClassInformation.get(superClassURI).toString
().indexOf("#") + 1;
                                                            if
(auxSuperClassName >= 0)
                                                            {

      String                contentSuperClassName                 =
(hashTableForEquivalentClassInformation.get(superClassURI)).toStri
ng().substring(auxSuperClassName);

      contentSuperClassName.trim();

      hashTableForSuperClassInformation.put(contentSuperClassName
+ "***superClasse***" + className, contentSuperClassName);
                                                            }
                                                      } else {
                                                            //NAO
TEM A INFORMACAO DE CLASSE EQUIVALENTE PARA A SUPER CLASSE

      hashTableForSuperClassInformation.put(superClassName       +
"***superClasse***" + className, superClassName);
                                                      }
                                                } else {
                                                      //TEM      A
INFORMACAO DE CLASSE EQUIVALENTE
                                                      if
(hashTableForEquivalentClassInformation.get(hashTableForEquivalent
ClassInformation.get(classURI))!= null)
                                                      {
                                                            //TEM
A INFORMACAO DE CLASSE EQUIVALENTE NAS DUAS ONTOLOGIAS

      //Removendo as informacaos da URI e pegando apenas o nome
"cru" da Classe .
                                                            int
auxClassName                                                     =
hashTableForEquivalentClassInformation.get(classURI).toString().in
dexOf("#") + 1;
                                                            if
(auxClassName >= 0)
                                                            {

      String                 equivalentClassName                 =
(hashTableForEquivalentClassInformation.get(classURI)).toString().
substring(auxClassName);

      equivalentClassName.trim();
      //SUBSTITUI  O  NOME  DA  CLASSE  PELO  NOME  DA  SUA  CLASSE
EQUIVALENTE

      hashTableForSuperClassInformation.put(superClassName        +
"***superClasse***" + equivalentClassName, superClassName);
                                                            }

      hashTableForEquivalentClassInformation.remove(hashTableForEq
uivalentClassInformation.get(classURI));

      //hashTableForEquivalentClassInformation.remove(classeAndURI
);
                                                      } else {
                                                            //TEM
A INFORMACAO DE CLASSE EQUIVALENTE EM UMA SO ONTOLOGIA
                                                            if
(hashTableForEquivalentClassInformation.get(superClassURI)!= null)
                                                            {

      //TEM A INFORMACAO DE CLASSE EQUIVALENTE PARA A SUPER CLASSE


      int                    auxSuperClassName                    =
hashTableForEquivalentClassInformation.get(superClassURI).toString
().indexOf("#") + 1;

      if (auxSuperClassName >= 0)

      {

      String                contentSuperClassName                 =
(hashTableForEquivalentClassInformation.get(superClassURI)).toStri
ng().substring(auxSuperClassName);

      contentSuperClassName.trim();

      hashTableForSuperClassInformation.put(contentSuperClassName
+ "***superClasse***" + className, contentSuperClassName);

      }
```

```
                                                                  } else
{

      //TEM A INFORMACAO DE CLASSE EQUIVALENTE EM UMA SO ONTOLOGIA
E NAO TEM A INFORMACAO DE CLASSE EQUIVALENTE PARA A SUPER CLASSE

      int                    auxClassName                    =
hashTableForEquivalentClassInformation.get(classURI).toString().in
dexOf("#") + 1;

      if (auxClassName >= 0)

      {

      String                   contentName                   =
(hashTableForEquivalentClassInformation.get(classURI)).toString().
substring(auxClassName);

      contentName.trim();



      //SUBSTITUI  O  NOME  DA  CLASSE  PELO  NOME  DA  SUA  CLASSE
EQUIVALENTE

      hashTableForSuperClassInformation.put(superClassName       +
"***superClasse***" + contentName, superClassName);

      }
                                                            }
                                                      }
                                                }

                                                //NAO     TEM     A
INFORMACAO DO NOME DA SUPERCLASSE
                                          }        else        if
(hashTableForEquivalentClassInformation.get(classURI) == null)
                                          {

      hashTableForSuperClassInformation.put("***semPai***"       +
"***superClasse***" + className, "***semPai***");
```

```
                                                 //TEM A INFORMACAO
DE CLASSE EQUIVALENTE
                                           } else
                                           {
                                                 int auxClassName =
(hashTableForEquivalentClassInformation.get(classURI)).toString().
indexOf("#") + 1;
                                                 if   (auxClassName
>= 0)
                                                 {
                                                       String
contentName                                                       =
(hashTableForEquivalentClassInformation.get(classURI)).toString().
substring(auxClassName);

      contentName.trim();

                                                       //SUBSTITUI
O NOME DA CLASSE PELO NOME DA SUA CLASSE EQUIVALENTE

      hashTableForSuperClassInformation.put("***semPai***"       +
"***superClasse***" + contentName, "***semPai***");
                                                 }
                                           }
                                     }

                               }         else          if          (
hashTableForEquivalentClassInformation.get(className) == null )
                               //Classe nao tem nome da SuperClasse
e nao tem classe equivalente cadastrada
                               {

      hashTableForSuperClassInformation.put("***semPai***"       +
"***superClasse***" + className, "***semPai***");
                               } else {
                                     int       auxClassName       =
(hashTableForEquivalentClassInformation.get(classURI)).toString().
indexOf("#") + 1;
                                     if (auxClassName >= 0)
                                     {
                                          String contentName =
(hashTableForEquivalentClassInformation.get(classURI)).toString().
substring(auxClassName);
                                          contentName.trim();

                                          //SUBSTITUI O NOME DA
CLASSE PELO NOME DA SUA CLASSE EQUIVALENTE

      hashTableForSuperClassInformation.put("***semPai***" +
"***superClasse***" + contentName, "***semPai***");
                                    }
                              }
                        }
                  }
            }
                  return hashTableForSuperClassInformation;
      }

      //Funcao responsável pela inserção em uma estrutura de dados
auxiliar as informações de classes e
      //classes equivalentes de uma dada ontologia.  Armazena os
namespaces e os nomes das classes.
      public static Hashtable equivalentClassInformation(OntModel
ontModelObj) {
            Hashtable hashTableForEquivalentClassInformation = new
Hashtable();

            String classNameAndURI = new String();
            String equivalentClassNameAndURI = new String();

            for (Iterator iteratorA = ontModelObj.listClasses();
iteratorA.hasNext();)
            {
                  OntClassImpl ont = (OntClassImpl)
iteratorA.next();

                  //Recuperando as classes que possuem a
informacao de classe equivalente
                  if (ont.getLocalName() != null)
                  {
                        classNameAndURI = ont.getURI();
                        classNameAndURI.trim();
                  }

                  if (ont.getEquivalentClass() != null) {
                        equivalentClassNameAndURI                =
ont.getEquivalentClass().getURI();
                        equivalentClassNameAndURI.trim();

      hashTableForEquivalentClassInformation.put(classNameAndURI,
equivalentClassNameAndURI);
                  }
            }
            return hashTableForEquivalentClassInformation;
      }


      //FUNCAO QUE ADICIONARA AS ULTIMAS INFORMACOES DE CLASSES
EQUIVALENTES, CASO ESTAS, AINDA NAO FORAM INCLUIDAS.
      public   static   Model   LastUpdateEquivalentClass(OntModel
ontModelObjVar1,       OntModel       ontModelObjVar2,       String
nameSpaceFirstOnt, String nameSpaceSecondOnt)
      {
            String auxLine = new String();
            String className = new String();
            String equivalentClassName = new String();

            System.out.println("***   ALINHAMENTOS   DO   TREEDIFF:
***");

            try {
                  BufferedReader auxFile = new BufferedReader(new
FileReader("treeDiffResults.txt"));

                  auxLine = auxFile.readLine();
                  while (auxLine != null)
                  {
                        int auxLineNumber = auxLine.indexOf("-");
                        if (auxLineNumber >= 0)
                        {
                              className   =   auxLine.substring(0,
auxLineNumber);
```

```
                              equivalentClassName                  =
auxLine.substring(auxLineNumber + 1);
                              System.out.println("className:  "  +
className + "--- equivalentClassName: " + equivalentClassName);

                              //A CLASSE NAO FOI IDENTIFICADA COMO
EQUIVALENTE NAS ETAPAS ANTERIORES
                              if                                   (
(hashTableForEquivalentClassInformation.get(nameSpaceFirstOnt     +
className) == null) ||

(hashTableForEquivalentClassInformation.get(nameSpaceSecondOnt    +
className) == null) )
                              {
                                    OntClass aClass = null;

                                    //PERCORRE      A      PRIMEIRA
ONTOLOGIA PARA SABER SE A CLASSE EQUIVALENTE A PERTENCE
                                    Iterator      iteratorA      =
ontModelObjVar1.listClasses();
                                    while (iteratorA.hasNext())
                                    {
                                          //PERCORRE  CADA  CLASSE
DA PRIMEIRA ONTOLOGIA
                                          OntClassImpl
ontObjFirstOntology = (OntClassImpl) iteratorA.next();
                                          if                   (
(ontObjFirstOntology.getNameSpace() != null) &&

(ontObjFirstOntology.getNameSpace()).equals(nameSpaceFirstOnt) )
                                          {
                                                //HA A NECESSIDADE
DESSA VERIFICACAO PARA ONTOLOGIAS QUE IMPORTAM OUTRAS.

                                                if             (
(ontObjFirstOntology.getLocalName()!= null) &&
                                                     (
(ontObjFirstOntology.getLocalName().trim()).equals(className) ) )
                                                {
                                                      aClass      =
ontModelObjVar2.createClass(nameSpaceSecondOnt                     +
equivalentClassName);

      aClass.addEquivalentClass(ontObjFirstOntology);
                                                }
                                          }
                                    }

                                    //PERCORRE A SEGUNDA ONTOLOGIA
PARA SABER SE A CLASSE EQUIVALENTE A PERTENCE
                                    Iterator      iteratorB       =
ontModelObjVar2.listClasses();
                                    while (iteratorB.hasNext())
                                    {
                                          //PERCORRE  CADA  CLASSE
DA PRIMEIRA ONTOLOGIA
                                          OntClassImpl
ontObjSecondOntology = (OntClassImpl) iteratorB.next();
                                          if
((ontObjSecondOntology.getNameSpace()       !=       null)        &&
(ontObjSecondOntology.getNameSpace()).equals(nameSpaceSecondOnt))
                                          {
                                                if                 (
(ontObjSecondOntology.getLocalName()!= null) &&
                                                     (
(ontObjSecondOntology.getLocalName().trim()).equals(className) ) )
                                                {
                                                      aClass      =
ontModelObjVar1.createClass(nameSpaceFirstOnt                      +
equivalentClassName);

      aClass.addEquivalentClass(ontObjSecondOntology);
                                                }
                                          }
                                    }
                              }
                        }
                        auxLine = auxFile.readLine();
                  }
                  auxFile.close();
```

```
        } catch (IOException e)
        {
            e.printStackTrace();
        }
        return ontModelObjVar2.union(ontModelObjVar1);
    }

    //FUNCAO QUE IMPRIMI ALGUMAS INFORMACOES DAS DUAS ONTOLOGIAS
NA CONSOLE DO AMBIENTE
    public static void mostraInfoDaOnto(OntModel ontModelObj) {

        System.out.println("");
        System.out.println("");
        System.out.println("***INFORMACOES DAS CLASSES
EQUIVALENTES DAS ONTOLOGIAS:***");

        for (Iterator iteratorA =
ontModelObj.listClasses();iteratorA.hasNext();)
        {
            OntClassImpl ont = (OntClassImpl)
iteratorA.next();

            if (ont.getEquivalentClass() != null) {
                System.out.println("URI da Classe
Equivalente: " + ont.getEquivalentClass());

                if (ont.getLocalName() != null) {

                    System.out.println("URI da Classe:"
+ ont.getURI());
                    System.out.println("");
                }
            }
        }
    }

    public static void main(String[] args) throws IOException {

        //TEM QUE CHAMAR NOVAMENTE PORQUE AO CONSTRUIR A
ESTRUTURA HIERARQUICA EM XML, ALGUM NOME DE CLASSE
```

```
            //EQUIVALENTE QUE EXISTIA NAS DUAS ONTOLOGIAS PODE TER
SIDO APAGADO.
            hashTableForEquivalentClassInformation = null;
            hashTableForEquivalentClassInformation =
equivalentClassInformation(leOnto("mappingOnto.owl"));

            //RECUPERA OS ARQUIVOS "CONGELADOS" DA PRIMEIRA ETAPA.

            Model ontModelObjFinalResult =
LastUpdateEquivalentClass(leOnto("newFirstOnto.owl"),
leOnto("newSecondOnto.owl"), "file:firstOnto.owl#",
"file:secondOnto.owl#");
            //ARMAZENA A INFORMACAO DOS DOIS MODELOS DAS DUAS
ONTOLOGIAS DE ENTRADA.

            //ADICIONA AS IMPORTACOES NECESSARIAS PARA EFETIVAR O
ALINHAMENTO.  SEM ISSO, SO TEM OS NAMESPACES SEM A IDENTIFICACAO
DO SEU RECURSO (ARQUIVO OU HTTP).
            Model modelAux = leOnto("cabecalho.owl");
            ontModelObjFinalResult =
ontModelObjFinalResult.add(modelAux);

            //Escreve um arquivo do tipo *.owl das duas ontologia
com seus nós Enriquecidos;
            File saidaFinalOnto = new
File("solCombPropFinalFile.owl");
            saidaFinalOnto.createNewFile();
            FileWriter outFinal = new FileWriter(saidaFinalOnto);
            ontModelObjFinalResult.write(outFinal, "RDF/XML-
ABBREV");

            mostraInfoDaOnto(leOnto("solCombPropFinalFile.owl"));
      }
    }
```