REGINA CLAUDIA CANTELE

**CONSTRUINDO ONTOLOGIAS A PARTIR DE RECURSOS
EXISTENTES: UMA PROVA DE CONCEITO NO DOMÍNIO DA
EDUCAÇÃO**

**São Paulo
2009**

REGINA CLAUDIA CANTELE

**CONSTRUINDO ONTOLOGIAS A PARTIR DE RECURSOS
EXISTENTES: UMA PROVA DE CONCEITO NO DOMÍNIO DA
EDUCAÇÃO**

Tese apresentada à Escola
Politécnica da Universidade de São
Paulo para a obtenção do título de
Doutor em Engenharia Elétrica.

**Área de Concentração:
Sistemas Digitais**

**Orientador:
Prof. Dr^a. Maria Alice Grigas Varella Ferreira**

**São Paulo
2009**

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 07 de maio de 2009.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Cantele, Regina Claudia

Construindo ontologias a partir de recursos existentes: uma prova de conceito no domínio da educação / R.C. Cantele. -- ed.rev. -- São Paulo, 2009. 226 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1. Semântica (Educação) 2. Ontologia 3. Engenharia de software I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II. t.

DEDICATÓRIA

Aos meus queridos pais Aparício e Eny pelo exemplo de vida e
à minha irmã Adriana.

AGRADECIMENTOS

Aos meus pais e minha irmã que incondicionalmente me apóiam, incentivam e acreditam nas minhas decisões de vida. Vocês são meu porto seguro.

A minha orientadora Maria Alice Grigas Ferreira por propiciar meu envolvimento no seu mundo de sabedoria, ética e fé.

À Diana Adamatti pelo interesse constante no progresso do meu trabalho, pelo incentivo e compartilhamento na escrita de vários artigos e pela amizade irrestrita.

Ao Moyses Araújo por ter aberto as portas do doutorado para mim e ser esta pessoa confiável e erudita.

Aos professores doutores da Escola Politécnica em especial Romero Tori, Graça Bressan e Jaime Sichman pelas valiosas sugestões e apoio teórico e técnico.

Aos administradores dos sistemas COL e Teleduc.

Aos meus amigos “sem os quais eu nada seria” e que com seus sorrisos, palavras de incentivo e carinho tornaram minha caminhada mais suave – Leticia, Silvia, Bia, Tatiana, Beto, Isacco, Marcelo Otavio, Marilan, Stael, Vidal, Lima e Ana Helena. Vocês confirmaram as palavras de Santo Agostinho: “Existem pessoas através das quais Deus me amou”.

Enfim, agradeço a todos àqueles que, desde o início, confiaram no meu esforço e dedicação, e contribuíram para a realização deste trabalho de alguma forma.

“Deus me deu o dom de não ter medo do amanhã
por acreditar no hoje”
Frei Jaime Bettega

RESUMO

Na Grécia antiga, Aristóteles (384-322 aC) reuniu todo conhecimento de sua época para criar a Enciclopédia. Na última década surgiu a Web Semântica representando o conhecimento organizado em ontologias. Na Engenharia de Ontologias, o Aprendizado de Ontologias reúne os processos automáticos ou semi-automáticos de aquisição de conhecimento a partir de recursos existentes. Por outro lado, a Engenharia de Software faz uso de vários padrões para permitir a interoperabilidade entre diferentes ferramentas como os criados pelo *Object Management Group* (OMG) – *Model Driven Architecture* (MDA), *Meta Object Facility* (MOF), *Ontology Definition Metamodel* (ODM) e *XML Metadata Interchange* (XMI). Já o *World Wide Web Consortium* (W3C) disponibilizou uma arquitetura em camadas com destaque para a *Ontology Web Language* (OWL). Este trabalho propõe um *framework* para reunir estes conceitos fundamentado no ODM, no modelo OWL, na correspondência entre metamodelos, nos requisitos de participação para as ferramentas e na seqüência de atividades a serem aplicadas até obter uma representação inicial da ontologia. Uma prova de conceito no domínio da Educação foi desenvolvida para testar esta proposta.

Palavras-chave: Web Semântica. Aprendizado de Ontologias. Metamodelos. Educação.

ABSTRACT

In ancient Greece, Aristotle (384-322 BCE) endeavored to collect all the existing science in his world to create the Encyclopedia. In the last decade, Berners-Lee and collaborators idealized the Web as a structured repository, observing an organization they called Semantic Web. Usually, domain knowledge is organized in ontologies. As a consequence, a great number of researchers are working on method and technique to build ontologies in Ontology Engineering. Ontology Learning meets automatic or semi-automatic processes which perform knowledge acquisition from existing resources. On the other hand, software engineering uses a collection of theories, methodologies and techniques to support information abstraction and several standards have been used, allowing interoperability and different tools promoted by the Object Management Group (OMG) – Model Driven Architecture (MDA), Meta Object Facility (MOF), Ontology Definition Metamodel (ODM) and XML Metadata Interchange (XMI). The World Wide Web Consortium (W3C) released architecture in layers for implementing the Semantic Web with emphasis on the Web Ontology Language (OWL). A framework was developed to combine these concepts based on ODM, on OWL model, the correlation between metamodels, the requirements for the tools to participate; in it, the steps sequence was defined to be applied until initial representations of ontology were obtained. A proof of concept in the Education domain was developed to test this proposal.

Keywords: *Semantic Web. Ontology Learning. Metamodels. Education.*

LISTA DE ILUSTRAÇÕES

Figura 1	- Servidor Educacional	25
Figura 2	- Arquitetura da Web Semântica	43
Figura 3	- Atividades da Engenharia de Ontologias	47
Figura 4	- Modelagem de Ontologia no Contexto MDA e WS	53
Figura 5	- Camadas de Elementos da Ontologia	61
Figura 6	- Arquitetura Conceitual	62
Figura 7	- Ontologias para Objetos de Aprendizagem	78
Figura 8	- Blocos Importantes da WSE	87
Figura 9	- Modelo de Referência Arquitetural	90
Figura 10	- Classificação dos LMS em Gerações	93
Figura 11	- Representação dos Metamodelos no <i>Framework</i>	99
Figura 12	- Visão Geral do <i>Framework</i> Proposto	109
Figura 13	- Diagrama de Atividades do <i>framework</i> proposto.....	122
Figura 14	- Ontologias, Usabilidade e Reusabilidade.....	125
Figura 15	- Metamodelo da Ferramenta Protégé	135
Figura 16	- Subconjunto do Script de Criação SQL.....	140
Figura 17	- Subconjunto do Diagrama de Classes.....	141
Figura 18	- Subconjunto do arquivo XML	142
Figura 19	- Modelo de Dados – Teleduc	143
Figura 20	- Detalhes do Modelo Conceitual do IMS LD	144
Figura 21	- Algoritmos da ferramenta Text2Onto	145
Figura 22	- Classe “ <i>Activity-structure</i> ” obtida a partir do <i>framework</i> proposto	150
Figura 23	- Classe “ <i>Activity-structure</i> ” do projeto EUME	152
Figura 24	- Edição da propriedade “ <i>execution-entity-ref</i> ”	154
Figura 25	- Classes e propriedades da representação inicial obtida para o COL	158
Figura 26	- Arquitetura para Ontologias no Domínio da Educação	167
Figura B1	- Diagrama da Descrição da Classe OWL	201
Figura B2	- Diagrama das Restrições de Classes OWL	202
Figura B3	- Exemplo Estereótipo	203
Figura B4	- Exemplo de Mapeamento UML para OWL	204
Figura B5	- Esquema Genérico de Transporte de Modelos ..	204
Figura D1	- Modelo conceitual do IMS LD	213
Figura E1	- Fragmento do Diagrama de Atividades do Módulo II do CCG	223
Figura E2	- Módulo II do CCG no Reload Player	226

LISTA DE QUADROS

Quadro 1	- Tipos de Ontologias	40
Quadro 2	- Metodologias com Abordagem no Desenvolvimento Centralizado	49
Quadro 3	- Metodologias com Abordagem no Desenvolvimento Colaborativo	49
Quadro 4	- Gerações de Metodologias	50
Quadro 5	- Abordagens MDA e Ontologias	55
Quadro 6	- Comparativo das Ferramentas para Ontologia	57
Quadro 7	- Ferramentas para Avaliação de Ontologias	58
Quadro 8	- Comparativo dos Sistemas de Aprendizado	67
Quadro 9	- Dimensões e Problemas do Ensino e Aprendizagem	72
Quadro 10	- Tipos de Aprendizagem e Uso de Novas Tecnologias	74
Quadro 11	- Web Semântica na Educação e suas expectativas	75
Quadro 12	- Comunidades e Metadados Propostos	83
Quadro 13	- Estrutura para Ontologia na Educação	86
Quadro 14	- Principais LORs	94
Quadro 15	- Elementos OWL	102
Quadro 16	- Correspondência entre Metamodelos E/R, Relacional e OWL	105
Quadro 17	- Correspondência entre Metamodelos UML e OWL	107
Quadro 18	- Correspondência entre Metamodelos LOM e OWL	108
Quadro 19	- Requisitos p/ Sistemas Existentes	113
Quadro 20	- Requisitos p/ Ferramentas CASE	114
Quadro 21	- Requisitos p/ Ferramentas AO a partir de Textos	116
Quadro 22	- Requisitos p/ Ferramentas de Transformação entre Metamodelos	118
Quadro 23	- Requisitos p/ Ferramentas do Ambiente de Ontologias	120
Quadro 24	- Sistemas Educacionais e requisitos do <i>framework</i> proposto.....	130
Quadro 25	- CASE e requisitos	132
Quadro 26	- Protégé e requisitos	134
Quadro 27	- Transformações e requisitos	137
Quadro 28	- Expressividade da representação OWL	155
Quadro A1	- Ferramentas para Anotação Manual	196
Quadro A2	- Ferramentas para Anotação Automática	196
Quadro A3	- Ferramentas para Anotação Integrada	197
Quadro A4	- Linguagens para representação de ontologias	197
Quadro A5	- Linguagens para Consultas de Ontologias	198
Quadro B1	- Exemplo de Transformação de Metamodelos	205
Quadro C1	- Resumo Elemento versus Técnica Utilizada no AO	209
Quadro D1	- Ferramentas para Edição IMS LD	215
Quadro E1	- Narrativa do Curso de Computação Gráfica	220
Quadro E2	- Verbos para atividades do aluno	222
Quadro E3	- Verbos para atividades dos professores	222
Quadro E4	- Siglas para os recursos de entrada e saída das atividades	223

LISTA DE TABELAS

Tabela 1	- Quantidade de elementos aprendidos.....	149
Tabela 2	- Expressividade das Representações	156

LISTA DE ABREVIATURAS E SIGLAS

A-CDF	<i>ARIADNE Course Description Format</i>
ACM	<i>Association for Computing Machinery</i>
ADL	<i>Advanced Distributed Learning</i>
ADLib	<i>Athabasca University Digital Library</i>
ADM	<i>Architecture-Driven Modernization</i>
AEON	<i>Automatic Evaluation of Ontologies</i>
AeroSWARM	<i>AeroText Semantic Web Automated Relation Markup</i>
AGR	<i>AICC Guidelines and Recommendations</i>
AICC	<i>Aviation Industry CBT (Computer-Based Training) Committee</i>
AIFB	<i>Institute of Applied Informatics and Formal Description Methods</i>
ALOCOM	<i>Abstract Learning Object Content Model</i>
AMG	<i>Automatic Metadata Generation</i>
ANSI	<i>American National Standards Institute</i>
AO	<i>Aprendizado de Ontologias</i>
API	<i>Application Program Interface</i>
ARIADNE	<i>Alliance of Remote Instructional Authoring and Distribution Networks for Europe</i>
ASK-LDT	<i>Advanced e-Services for the Knowledge Society Learning Designer Toolkit</i>
ASTM	<i>Abstract Syntax Tree Meta-Model</i>
ATL	<i>Atlas Transformation Language</i>
BOTL	<i>Basic Object-oriented Transformation Language</i>
CAREO	<i>Campus Alberta Repository of Educational Objects</i>
CASE	<i>Computer Aided Software Engineering</i>
CBT	<i>Computer-Based Training</i>
CeLeBraTe	<i>Context e-Learning with Broadband Technologies</i>
CEN	<i>European Committee for Standardization</i>
CGXML	<i>Conceptual Graphs XML</i>
CIM ₁	<i>Computer Independent Model</i>
CIM ₂	<i>Common Information Model</i>
CL	<i>Common Logic</i>
CLOS	<i>Common Lisp Object System</i>
CLOE	<i>Co-Operative Learning Object Exchange</i>
CMF	<i>Common Media Framework</i>
CML	<i>Conceptual Modelling Language</i>
CMM-i	<i>Capability Maturity Model integration</i>
CMS	<i>Content Management Systems</i>
COHSE	<i>Conceptual Open Hypermedia Services Environment</i>
CP	<i>Content Packaging</i>
CREAM	<i>CREating Relational Metadata</i>

CSCLS	<i>Computer Supported Collaborative Learning Systems</i>
DAML	<i>DARPA Agent Markup Language</i>
DAML+OIL	<i>DARPA Agent Markup Language</i>
DBMS	<i>Database Management Systems</i>
DCMI	<i>Dublin Core Metadata Initiative</i>
DL	<i>Description Logics</i>
DODDLE	<i>Domain Ontology Rapid Development Environment</i>
DoD	<i>Departament of Defense</i>
DOE	<i>Differential Ontology Editor</i>
DTD	<i>Document Type Definition</i>
DUET	<i>DAML UML Enhanced Tool</i>
EAD	<i>Educação a Distância</i>
EDNA	<i>Education Network Austrália</i>
EMF	<i>Eclipse Modelling Framework</i>
EML	<i>Educational Modeling Language</i>
EO	<i>Engenharia de Ontologias</i>
EOE	<i>Envolving Ontology Engineering</i>
E/R	<i>Entidade/Relacionamento</i>
eRIB	<i>eduSource Repository in a Box</i>
ES	<i>Engenharia de Software</i>
EUME	<i>Ubiquitous and Multimedia Enviroment for Education</i>
F-Logic	<i>Frame Logic</i>
FAPESP	<i>Fundação de Apoio à Pesquisa do Estado de São Paulo</i>
FCA	<i>Formal Concept Analysis</i>
FOIL	<i>Learning Sets of First-Order Rules</i>
FOML	<i>Formal Ontology Markup Language</i>
GATE	<i>General Architecture for Text Engineering</i>
GXL	<i>Graph Exchange Language</i>
HCOME	<i>Human-Centered Ontology Engineering Methodology</i>
HTML	<i>Hyper Text Markup Language</i>
HTTP	<i>HyperText Transfer Protocol</i>
IDLD	<i>Implementation and Deployment of the Learning Design</i>
IEC	<i>International Electrotechnical Commission</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
IMAT	<i>Integrating. Materials and Training</i>
IMS	<i>Instructional Management System Global Learning Consortium</i>
IMS LD	<i>IMS Learning Design</i>
INTERLAB	<i>Laboratório de Tecnologias Interativas</i>
IRI	<i>Internationalized Resource Identifiers</i>
ISO	<i>International Standardization Organization</i>
KAON	<i>Karlsruhe Ontology and Semantic Web Infrastructure</i>

KDM	<i>Knowledge Discovery Metamodel</i>
KEA	<i>Keyphrases Extraction Algorithm</i>
KIF	<i>Knowledge Interchange Format</i>
KIM	<i>Knowledge and Information Management</i>
KPS	<i>Knowledge Pool System</i>
LAMS	<i>Learning Activity Management System</i>
LARC	Laboratório de Arquitetura e Redes de Computadores
LD	<i>Learning Design</i>
LDA	Laboratório de Desenvolvimento Associado
LGR	<i>Lydia Global Repository</i>
LIP	<i>Learner Information Package</i>
LMML	<i>Learning Material Mark-up Language</i>
LMS	<i>Learning Management Systems</i>
LN4LD	<i>Learning Network for Learning Design</i>
LOCO	<i>Learning Object Context Ontology</i>
LOM	<i>Learning Object Metadata</i>
LOM	<i>Lexicon-based Ontology Mapping</i>
LOM	<i>Learned Ontology Model</i>
LOR	<i>Learning Object Repository</i>
LQS	<i>Lexicon Query Service</i>
LSD	<i>Learning Source Descriptions</i>
LTG	<i>Language Technology Group</i>
LSI	<i>Latent Semantic Indexing</i>
LTSC	<i>Learning Technology Standards Committee</i>
M0	instância do Modelo
M1	Modelo
M2	Metamodelo
M3	Meta-metamodelo
MAFRA	<i>MApping Framework</i>
MAGIC	<i>Metadata Automated Generation for Instrucional Content</i>
MDA	<i>Model Driven Architecture</i>
MDE	<i>Model-Driven Engineering</i>
MDR	<i>Metadata Repository</i>
MERLOT	<i>Multimedia Educational Resource for Learning and Online Teaching</i>
MIWeb	<i>Mediator-based Integration of Web Sources</i>
MIT	<i>Massachusetts Institute of Technology</i>
MOF	<i>Meta Object Facility</i>
MOMIS	<i>Mediator Environment for Multiple Information Sources</i>
MPEG	<i>Moving Picture Experts Group</i>
MPL	<i>Modeling Primitive Library</i>
NASA	<i>National Aeronautics and Space Administration</i>

NIME	<i>National Institute of Multimedia Education</i>
NKRL	<i>Narrative Knowledge Representation Language</i>
nRQL	<i>new RACER Query Language</i>
NSDL	<i>National Science Digital Library</i>
OA	Objetos de Aprendizagem
OAU	Ontologia de Aplicação da Unidade de Aprendizagem
OCL	<i>Object Constraint Language</i>
OCML	<i>Operational Conceptual Modeling Language</i>
OD	Ontologia de Domínio
ODE ₁	<i>Ontology Development Process</i>
ODE ₂	Ontologia do Domínio da Educação
ODLD	Ontologia de Domínio do <i>Learning Design</i>
ODM	<i>Ontology Definition Metamodel</i>
OEC	Ontologia de estrutura do conteúdo
OIL	<i>Ontology Inference Layer</i>
OKBC	<i>Open Knowledge Base Connectivity</i>
OKI	<i>Open Knowledge Initiative</i>
OM	Ontologia de Metadados
OMG	<i>Object Management Group</i>
OML	<i>Ontology Markup Language</i>
ONES	<i>One-Stop eLearning Portal</i>
ONION	<i>ONtology compositiON</i>
ONIONS	<i>ONtological Integration Of Naive Sources</i>
OOF	<i>Open Ontology Forge</i>
OTK	<i>On-To-Knowledge</i>
OUNL	<i>Open Universiteit Nederland</i>
OUP	<i>Ontology UML Profile</i>
OWL	<i>Ontology Web Language</i>
OWL DL	<i>OWL Description Logic</i>
OWL-QL	<i>OWL Query Language</i>
P2P	<i>peer-to-peer</i>
PA	Projeto de Aprendizagem
PAL	<i>Protégé Axiom Language</i>
PANKOW	<i>Pattern-based Annotation Knowledge through On the Web</i>
PAPI	<i>Personal and Private Information Standard</i>
PIDS	<i>Person Identification Service</i>
PILO	<i>Practioner Inquiry Learning Object</i>
PIM	<i>Platform Independent Model</i>
PLN	Processamento de Linguagem Natural
POOL	<i>Portal for Online Objects in Learning</i>
POM	<i>Probabilistic Ontology Model</i>

POPS	<i>People, Organizations, Projects, Skills</i>
PROMETEUS	<i>PRoMoting Multimedia access to Education and Training in the European Society</i>
PLSI	<i>Probabilistics LSI</i>
PSM	<i>Platform Specific Model</i>
QTI	<i>Question & Test Interoperability</i>
QVT	<i>Query View Transformations</i>
RAINBOW	<i>Reusable Architecture for Intelligent Brokering Of Web</i>
RAMLET	<i>Resource Aggregation Model for Learning, Education and Training</i>
RDF	<i>Resource Description Framework</i>
RDFS	<i>RDF Schema</i>
RDQL	<i>RDF Data Query Language</i>
RELOAD	<i>Reusable eLearning Object Authoring & Delivery</i>
RIF	<i>Rule Interchange Format</i>
ROSA	<i>Repository of Objects with Semantic Access</i>
RuleML	<i>Rule Modeling Language</i>
SCORM	<i>Sharable Content Object Reference Model</i>
S-CREAM	<i>Semi-automatic CREAtion of Metadata</i>
SEISD	<i>System for Extraction of Semantic Information from Dictionaries</i>
SeLeNe	<i>Self e-Learning Networks</i>
SGBDR	<i>Sistema Gerenciador de Banco de Dados Relacional</i>
SGML	<i>Standard Generalized Markup Language</i>
SHOE	<i>Simple HTML Ontology Extension</i>
SKOS	<i>Simple Knowledge Organization Systems</i>
SMETE	<i>Science Mathematics Engineering Technology Education</i>
SMORE	<i>Semantic Markup, Ontology, and RDF Editor</i>
SMU	<i>Semantically Meaningful Unit</i>
SOAT	<i>Semi-Automatic Domain Ontology Acquisition Tool</i>
SPARQL	<i>Simple Protocol and RDF Query Language</i>
SQL	<i>Structured Query Language</i>
SUO	<i>Standard Upper Ontology</i>
SWALEM	<i>Semantic Web Assisted Learning Experience Management</i>
SWRL	<i>Semantic Web Rule Language</i>
TAO	<i>Transitioning Applications to Ontologies</i>
TARGETEAM	<i>TArgeted Reuse and GEneration of TEaching Materials</i>
TELOS	<i>TeLEarning Operating System</i>
TFIDF	<i>Term frequency, inverse document frequency</i>
TI	<i>Tecnologia da Informação</i>
TIC	<i>Tecnologia, Informação e Comunicação</i>
TIDIA-Ae	<i>Tecnologia da Informação para o Desenvolvimento da Internet Avançada - Aprendizagem Eletrônica</i>
TILE	<i>The Inclusive Learning Exchange</i>

TM4L	<i>Topic Maps for e-Learning</i>
TML	<i>Tutorial Mark-up Language</i>
TOVE	<i>TOronto Virtual Enterprise</i>
ULTRA	<i>UREKA Learning-Object Taxonomy & Repository Architecture</i>
UML	<i>Unified Modeling Language</i>
UML2	<i>Unified Modeling Language version 2</i>
UoL	<i>Unit of Learning</i>
UPON	<i>Unified Process for Ontology Building</i>
URI	<i>Uniform Resource Identifier</i>
VOM	<i>Visual Ontology Modeler</i>
VOSSAT	<i>Visualized Online Simple Sequencing Authoring Tool</i>
VPR	<i>Virtual Private Repository</i>
W3C	<i>World Wide Web Consortium</i>
WickOffice	<i>Writing in the Context of Knowledge Office</i>
WOLFIE	<i>Word Learning From Interpreted Examples</i>
WS	<i>Web Semântica</i>
WSE	<i>Web Semântica na Educação</i>
XMI	<i>XML Metadata Interchange</i>
XML	<i>eXtensible Markup Language</i>
XOL	<i>XML-Based Ontology Exchange Language</i>
XSL	<i>Extensible Style Language</i>
XSLT	<i>XML Stylesheet Language Transformation</i>
XTM	<i>XML Topic Maps</i>
YATL	<i>Yet Another Transformation Language</i>

LISTA DE SÍMBOLOS

- $\neg$ *negação*
- $\wedge$ *conjunção*
- $\vee$ *disjunção*
- $\leftrightarrow$ *dupla implicação*
- $\rightarrow (\supset)$ *implicação*
- $\forall$ *quantificador universal*
- $\exists$ *quantificador existencial*
- $\subseteq$ *está contido*
- $\leq$ *menor igual*
- $\geq$ *maior igual*

SUMÁRIO

1	INTRODUÇÃO	22
1.1	Contextualização	22
1.2	Justificativas	25
1.3	Motivação	27
1.4	Objetivos	29
1.5	Contribuições	30
1.6	Metodologia	31
1.7	Organização do trabalho	33
2	WEB SEMÂNTICA E ONTOLOGIA	34
2.1	Cenário Atual	34
2.2	Metadados e Anotação Semântica	35
2.3	Ontologias	38
2.4	Arquitetura	42
2.5	Engenharia de Ontologias	46
2.5.1	Metodologia	46
2.5.2	Abordagem na Engenharia de Software	51
2.5.3	Ferramentas	55
2.6	Aprendizado de Ontologias	58
2.6.1	Processo Básico	59
2.6.2	Técnicas	63
2.6.3	Ferramentas	65
2.7	Discussão	69
3	EDUCAÇÃO E WEB SEMÂNTICA	71
3.1	Cenário Atual	71
3.1.1	Objetos de Aprendizagem	76
3.1.2	Projeto de Aprendizagem	79
3.2	Metadados e Ontologias Educacionais	81
3.3	Sistemas Educacionais	90
3.3.1	Desenvolvimento	91
3.3.2	Publicação	91
3.3.3	Entrega	92
3.4	Discussão	96
4	FRAMEWORK PARA APRENDIZADO DE ONTOLOGIAS	97
4.1	Definição Formal	98
4.2	Modelo OWL	101
4.3	Correspondência entre Metamodelos	102
4.3.1	Modelo de Dados	103
4.3.2	Diagrama de Classes	105
4.3.3	Textos	108
4.4	Visão Geral	109
4.4.1	Recursos do Domínio	109
4.4.2	Aprendizado de Ontologias	110
4.4.2.1	<i>Schematas</i>	110

4.4.2.2	A partir de Textos	111
4.4.3	Ambiente de Ontologias	111
4.4.4	Engenharia Progressiva de Ontologias	112
4.5	Requisitos para Participação	112
4.5.1	Sistemas Existentes	113
4.5.2	Ferramenta CASE	114
4.5.3	AO a partir de textos	116
4.5.4	Transformação de Metamodelos	118
4.5.5	Ambiente de Ontologias	119
4.6	Atividades do <i>framework</i> proposto	121
4.6.1	Selecionar Recursos do Domínio	122
4.6.2	Selecionar Ferramentas	123
4.6.3	Realizar o Aprendizado de Ontologias	124
4.6.4	Utilizar o Ambiente de Ontologias	124
4.6.5	Selecionar Método para Engenharia Progressiva	125
4.7	Discussão.....	126
5	PROVA DE CONCEITO NO DOMÍNIO DA EDUCAÇÃO	129
5.1	Aplicação do <i>Framework</i> Proposto	129
5.1.1	Selecionar Recursos do Domínio	130
5.1.2	Selecionar Ferramentas	131
5.1.2.1	CASE.....	131
5.1.2.2	AO a partir de textos.....	132
5.1.2.3	Ambiente de Ontologias.....	133
5.1.2.4	Transformação entre Metamodelos.....	136
5.1.3	Realizar Aprendizado de Ontologias	139
5.1.3.1	AO a partir de <i>Schematas</i>	139
5.1.3.2	AO a partir de Textos.....	145
5.1.4	Utilizar Ambiente de Ontologias	146
5.1.5	Selecionar Método para Engenharia Progressiva	147
5.2	Resultados Obtidos	149
5.2.1	Quantidade de Elementos Aprendidos	149
5.2.2	Expressividade da Representação OWL	154
5.2.3	Conhecimento do Engenheiro de Ontologias	157
5.2.4	Diferenças nas representações aprendidas	159
5.3	Discussão	160
6	CONSIDERAÇÕES FINAIS	161
6.1	Síntese do trabalho	161
6.2	Propostas de Trabalhos Futuros	167
	REFERÊNCIAS.....	169
	GLOSSÁRIO.....	187
	ANEXO A – Ferramentas para WS	196
	ANEXO B - <i>Ontology Definition Metamodel</i>	199
	ANEXO C - Métodos e técnicas para AO	206
	ANEXO D - Padrões de projeto de aprendizagem	211
	ANEXO E – Curso de Computação Gráfica	217

1 INTRODUÇÃO

Andrônico de Rodes, por volta de 50 a.C., recolheu e classificou as obras de Aristóteles (384-322 a.C) que, durante muitos séculos, haviam ficado dispersas e perdidas. Isto mostra que, desde este período, o homem deseja organizar o conhecimento coletado: pelas organizações, comunidades científicas e indivíduos em diferentes fontes de informação. Este trabalho também busca reunir conceitos existentes em diferentes áreas da Tecnologia da Informação (TI) para aplicá-los na Web Semântica na Educação (WSE). Este capítulo contém a contextualização do trabalho com as justificativas para o desenvolvimento do tema e a motivação que levaram à sua escolha. Inclui, ainda, os objetivos, as contribuições esperadas para a área de estudo, a metodologia adotada e a organização do trabalho.

1.1 Contextualização

Neste gigantesco mundo Web onde milhares de páginas são disponibilizadas a cada dia com conteúdos cada vez mais variados, apresentados em distintos formatos, e em diferentes contextos, a atuação das máquinas é necessário para, muito mais do que realizar uma pesquisa e retornar páginas similares, apresentar resultados de acordo com o significado semântico envolvido.

O conceito de semântica no mundo Web foi introduzido por Tim Berners-Lee e colaboradores (2001) e desencadeou o surgimento de novos processos, técnicas, linguagens e ferramentas para permitir o trabalho em cooperação entre máquinas e humanos. Assim surgiu a Web Semântica (WS), responsável por interligar significados de palavras e conteúdos publicados na Web. Este significado pode estar presente em metadados. Metadados são dados que descrevem propriedades de um recurso para diversos propósitos, como o contexto no qual o recurso se insere, sua qualidade, suas condições de uso, sua identificação, suas estratégias de preservação, entre outras.

Os metadados podem estar inseridos nos próprios recursos que descrevem ou armazenados em repositórios. Por exemplo, em páginas HTML (*Hyper Text Markup*

Language) é possível inserir metadados através de marcações não visíveis na exibição da página pelo navegador. Os metadados armazenados em repositórios aparecem na forma de anotações. Uma anotação é uma nota explicativa ou crítica anexada ao recurso. Anotação semântica identifica formalmente conceitos e relações entre conceitos nos recursos Web, para ser utilizada por máquinas para recuperação de informação e para garantia de interoperabilidade (UREN et al., 2006). Anotador e consumidores da anotação devem compartilhar o mesmo significado e para isto referências a uma ontologia se fazem necessárias.

O termo ontologia refere-se a um artefato de engenharia, constituído por um vocabulário específico para descrever um modelo particular do domínio, adicionando um conjunto explícito de suposições ou decisões, relacionando o significado das palavras no vocabulário (MAEDCHE, 2002). Uma ontologia envolve, então, um vocabulário de representação para capturar os conceitos e relações em algum domínio e um conjunto de axiomas, para restringir a sua interpretação (ANTONIOU; VAN HARMELEN, 2003).

A WS propõe anotar o conteúdo dos documentos usando informações semânticas de um domínio ontológico. O resultado são páginas interpretadas automaticamente pelos computadores através de agentes ou serviços Web Semânticos (OBERLE et al., 2005; FENSEL; BUSSLER, 2002).

Existem muitas linguagens para anotação de recursos e para representação e consultas a ontologias, e também muitas ferramentas para desenvolvimento, avaliação e manutenção das ontologias. O consórcio W3C (*World Wide Web Consortium*) coordena a WS apoiado pela indústria de software (W3C, 2008) e propõe uma arquitetura em camadas para construir aplicações na WS. Esta arquitetura define as tecnologias necessárias para que os conteúdos dos recursos Web possam ser compreendidos pelas máquinas e sua contribuição é providenciar um conjunto de padrões - linguagens, sintaxe comum, métodos.

A Engenharia de Ontologias (EO) surge para reunir estas linguagens e ferramentas e adicionar um conjunto de atividades importantes para o processo de desenvolvimento de ontologias e para o ciclo de vida da ontologia (PINTO; TEMPICH; SURE, 2004). Assim, a EO tem atividades incluídas nos processos de construção, aprendizagem, projeto, avaliação, validação, manutenção, integração, compartilhamento e reutilização de ontologias (SIMPERL; TEMPICH, 2006). Destaca-se neste trabalho o Aprendizado de Ontologia (AO): um conjunto de

métodos e técnicas usado para construir uma ontologia através de processos automáticos ou semi-automáticos de aquisição de conhecimento via textos, dicionários, bases de conhecimento, dados semi-estruturados e esquemas relacionais já existentes (SPYNS; MEERSMAN; JARRAR, 2002; GÓMEZ-PÉREZ; MANZANO-MACHO, 2003).

Alguns autores unem EO com a Engenharia de Software (ES) e abordam o uso da *Model Driven Architecture* (MDA) e seus padrões: *Meta-Object Facility* (MOF), *Unified Modeling Language* (UML) e *XML Metadata Interchange* (XMI), propostos pelo *Object Management Group* (OMG). O OMG¹ é um grupo dedicado a fornecer diretrizes para a indústria de software, através de especificações de padrões, de forma a facilitar a representação das aplicações Web, de integração de aplicações corporativas, de sistemas em tempo real e de plataformas distribuídas.

O domínio da Educação, como os demais domínios, pode inserir a WS no seu cotidiano para acompanhar as mudanças necessárias e a evolução do ambiente educacional. Em termos gerais, a mudança educacional deve aumentar a eficácia, a flexibilidade, a acessibilidade e a atratividade da educação, e diminuir a carga de trabalho da equipe de colaboradores, obtendo como resultado final a diminuição dos custos institucionais (KOPER, 2004). Também, conforme o tipo de aprendizagem proposto, como exposição dedutiva, pode ser interessante o uso de novas tecnologias (CAMPOS; DA ROCHA; DE CAMPOS, 1998).

Como consequência, necessita-se de novos métodos pedagógicos para suprir as exigências na educação e treinamento diário. Mais especificamente, propõe “sistemas baseados em semântica e no contexto para adquirir, organizar, personalizar, compartilhar e usar o conhecimento dentro da Web e, ainda, para conseguir a interoperabilidade semântica entre recursos da informação e serviços heterogêneos” (ARROYO et al., 2006).

A Web Semântica na Educação provê significado para representar, organizar e interconectar o conhecimento dos educadores, estudantes e autores de material educacional numa forma processável e compreensível por máquinas (DEVEDŽIĆ, 2006). Na Figura 1 observa-se a separação dos serviços educacionais - conteúdo educacional, personalização e serviços –, das ferramentas de autoria e das ferramentas de aprendizagem; a interação se dá através de agentes pedagógicos ou

¹ <http://www.omg.org/>

de requisições diretas. Para realização destes serviços, o Servidor Educacional faz uso de várias ontologias (O_1 a O_n , na Figura 1).

Figura 1 – Servidor Educacional, adaptada de (DEVEDŽIĆ, 2006)

Organismos internacionais já padronizaram metadados para objetos de aprendizagem, modelo de estudante e projeto de aprendizagem entre outros. Assim, muitas pesquisas aproveitam estes metadados para construir sua ontologia de referência na Educação.

Este trabalho trata da interdisciplinaridade entre três áreas: Engenharia de Ontologias, Engenharia de Software e Sistemas de Informação em Educação, propondo a utilização de padrões da MDA, associados aos padrões W3C para WS no AO, e de metadados educacionais para objetos de aprendizagem e projeto instrucional.

1.2 Justificativas

As justificativas para elaboração deste trabalho reúnem a necessidade de disponibilização de ontologias para o sucesso da WS, a existência de padrões definidos pelo W3C, o emprego de técnicas e ferramentas da ES na EO, a existência de representações de conhecimento nos sistemas existentes e a necessidade do

engenheiro de ontologias de um ambiente integrado para iniciar a construção de ontologias a partir de recursos já existentes.

A WS depende da interação complexa de muitas tecnologias envolvendo ontologias (OBERLE et al., 2005). O desenvolvimento de ontologias requer tempo, investimento e muito esforço (KIM;CHOI, 2007) pois é necessário disponibilizar muitas ontologias integradas num curto espaço de tempo, principalmente na WSE (DEVEDŽIĆ; JOVANOVIĆ; GAŠEVIĆ, 2007). No domínio da Educação, a utilização da WS é primordial para, principalmente, atender as novas demandas das instituições de ensino como o aprendizado contínuo ao longo da vida do indivíduo.

W3C padronizou as camadas e linguagens a serem utilizadas no universo da WS. Assim, as linguagens para representação de ontologias são RDFS (*Resource Description Framework Schema*) e OWL (*Ontology Web Language*) e, para deduções aplicáveis nas instâncias, a SWRL (*Semantic Web Rule Language*).

Na Ciência da Computação, semântica não é um termo novo e áreas de pesquisa tais como Recuperação e Extração de Informação, Lingüística Computacional, Inteligência Artificial e Gerenciamento de Dados já abordam ações relacionadas com a semântica nos seus campos de atuação há muito tempo. Estas diferentes abordagens implicam em muitas visões diferentes de cognição, de conceitos e de significados (SHETH, RAMAKRISHNAN; THOMAS, 2005). Cada área possui uma série de técnicas e ferramentas para obtenção dos muitos elementos da ontologia como conceitos, sinônimos e relações taxonômicas.

Considerando a representação da semântica formal, as abordagens para modelagem, ontologias e MDA iniciaram seu desenvolvimento em paralelo, mas por comunidades diferentes – Inteligência Artificial e Engenharia de Software –, respectivamente. Elas têm pontos e ações comuns. Muitos autores unem as diferenças existentes entre as abordagens e propõem soluções conjuntas para a engenharia progressiva de ontologias (GAŠEVIĆ; DJURIĆ; DEVEDŽIĆ, 2006; DJURIĆ; GAŠEVIĆ; DEVEDŽIĆ, 2005), principalmente, no uso de metamodelos padronizados entre as diferentes ferramentas utilizadas na Engenharia de Ontologias.

No domínio da Educação já existe uma variedade de sistemas para apoiar as atividades de ensino-aprendizagem nas escolas e universidades e para satisfazer as necessidades de aprendizagem do indivíduo ao longo da vida. Os principais tipos de sistemas são: Sistemas Gerenciadores de Conteúdo (*Content Management Systems*

– CMS), Sistemas Gerenciadores da Aprendizagem (*Learning Management Systems* – LMS) e Sistemas Apoiados por Computador para Aprendizagem Colaborativa (*Computer Supported Collaborative Learning Systems* – CSCLS). Os dois primeiros são, em princípio, orientados para o conteúdo, isto é, seu foco principal está na disciplina e atividades de aprendizagem. O último tipo é baseado nas experiências de aprendizagem social e usa a tecnologia para levar os estudantes a interagirem na construção do conhecimento (CARTELLI, 2006). Para promover o entendimento comum entre os sistemas educacionais e atingir um número maior de estudantes, com maior eficiência e eficácia, organismos internacionais de padronização buscam promover a interoperabilidade e reuso dos conteúdos pedagógicos pelo uso de metadados, para seus principais conceitos - objetos de aprendizagem e projeto de aprendizagem. Esses metadados compartilham um entendimento comum, via ontologias.

O engenheiro de ontologias está cercado de muitos recursos para representar as ontologias – métodos, técnicas, linguagens e ferramentas. Ele pode iniciar seu trabalho contando com a participação de especialistas do domínio ou pode obter representações iniciais da ontologia usando técnicas automáticas ou semi-automáticas, aplicadas nas representações de conhecimento existentes e, a partir deste ponto, buscar o refinamento, o compartilhamento e a aprovação do conhecimento representado.

Surge então a necessidade de integrar forças e convergir tecnologias para acelerar a construção de ontologias no domínio da Educação.

1.3 Motivação

A principal motivação é aproveitar os resultados das diversas pesquisas em Engenharia de Ontologias, Aprendizado de Ontologias, Web Semântica e Engenharia de Software, e disponibilizá-los para o engenheiro de ontologias num *framework* a fim de facilitar a construção de ontologias a partir do conhecimento já representado em sistemas existentes. Para isto é necessário o uso de padrões consolidados e aceitos tanto pela comunidade da Engenharia de Software como pela Engenharia de Ontologias.

Padrões são importantes, porque estabelecem, num alto nível de abstração, os princípios para os metamodelos das abstrações desejadas, organizando os modelos e o desenvolvimento de ferramentas que os representam. Regulam também a interoperabilidade entre as ferramentas e permitem o intercâmbio e o reuso de modelos. As pesquisas em Aprendizado de Ontologias já permitiram construir inúmeras ferramentas, representando diferentes técnicas para obtenção de ontologias e estas ferramentas, ao expressarem seus resultados em metamodelos padronizados, serão mais facilmente integradas a outras ferramentas para construção de ontologias, visto o sucesso já obtido em outras áreas como Banco de Dados, com os modelos E/R (Entidade/Relacionamento) e SQL (*Structured Query Language*). Desde a década de 80, estes dois metamodelos padronizados existem na grande maioria dos Sistemas Gerenciadores de Banco de Dados Relacionais (SGBDR), ferramentas CASE (*Computer Aided Software Engineering*) e linguagens de programação. Isto tem garantido a portabilidade, interoperabilidade, flexibilidade e reusabilidade dos modelos e das instâncias representadas em diferentes ambientes computacionais.

Padrões são também aplicados em metadados de domínio. Os metadados padronizados são especificados por comunidades e organismos interessados na troca de informações entre sistemas. No domínio da Educação existe o IMS (*Instructional Management System*) *Learning Design* (LD), representado de acordo com os princípios da Engenharia de Software e seus diagramas UML, que permite desde a autoria de projetos de aprendizagem até sua execução em diferentes LMS. Esse padrão tem sido recomendado pelo projeto de pesquisa TIDIA-Ae² (Tecnologia da Informação para o Desenvolvimento da Internet Avançada - Aprendizagem Eletrônica), financiado pela FAPESP (Fundação de Apoio à Pesquisa do Estado de São Paulo), cujo objetivo geral é construir um ambiente de ferramentas para *e-learning* voltado para aplicação no ensino superior, baseado em software livre e código aberto. O Interlab³ (Laboratório de Tecnologias Interativas) participa da fase II deste projeto como um Laboratório de Desenvolvimento Associado (LDA) e um dos seus objetivos é disponibilizar o Curso de Computação Gráfica neste ambiente, o que torna possível aplicar o padrão IMS LD neste trabalho.

² <http://tidia-ae.incubadora.fapesp.br/portal>

³ <http://www.pcs.usp.br/~interlab/>

Outra motivação importante é facilitar o entendimento e a construção de ontologias. Por ser uma área relativamente nova em tecnologia, os profissionais têm dificuldades em definir ontologia e distinguir entre tarefas para reuso de ontologias, aprendizado ou, mais genericamente, aquisição de conhecimento. Simperl e Tempich (2006) apontaram que somente um pequeno percentual dos projetos em estudo segue uma abordagem sistemática para construção de ontologia e poucos compartilham uma única metodologia.

Motivação adicional são os trabalhos similares como os de Zavitsanos, Vouros e Paliouras (2006), Maedche (2002), Djurić, Gašević e Devedžić (2005), Baclawski et al. (2002) e Cranefield e Pan (2007) e os *frameworks* existentes como KAON (*Karlsruhe Ontology and Semantic Web Infrastructure*), Asium, OntoLearn e OntoLT. Estes trabalhos serão melhor abordados nos capítulos 2 e 3.

1.4 Objetivos

O objetivo principal deste trabalho é propor um *framework* para Aprendizado de Ontologias a ser disponibilizado para o uso do engenheiro de ontologia, na sua atividade inicial de obtenção de uma primeira representação da ontologia, a partir de representações presentes em modelos de sistemas existentes. Este *framework* fará uso de diversas ferramentas construídas a partir das técnicas de AO e dos padrões de metamodelos e, será aplicado na construção de uma representação inicial de ontologia para o domínio da Educação.

Para atingir este objetivo pretende-se também:

- ☐ apresentar os conceitos, benefícios, fundamentos e padrões da WS considerados neste trabalho;
- ☐ abordar os métodos, técnicas e ferramentas existentes e aplicados na Engenharia de Ontologias;
- ☐ investigar arquiteturas, técnicas e ferramentas existentes para AO;
- ☐ detalhar os padrões existentes na Engenharia de Software para promover a interoperabilidade e compartilhamento de metamodelos entre diferentes ferramentas;
- ☐ aplicar os conceitos de meta-metamodelo definidos pela MDA/MOF nas ferramentas de AO e ambiente de ontologias;

- destacar aplicações e metadados padrões existentes no domínio da Educação;
- criar um *framework* para os engenheiros de ontologias utilizarem o AO;
- realizar uma prova de conceito do *framework* no domínio da Educação.

1.5 Contribuições

Este trabalho tem como principais contribuições:

- união dos processos e padrões existentes na Engenharia de Software aos processos e técnicas existentes na Engenharia de Ontologias, mais especificamente, ao Aprendizado de Ontologia;
- introdução dos conceitos de meta-metamodelo nas ferramentas do Aprendizado de Ontologia com seus respectivos mapeamentos;
- disponibilização de um *framework* padronizado para o engenheiro de ontologias utilizar nas suas atividades iniciais para construção de uma ontologia;
- inserção de um modelo para uso de ontologias no domínio da Educação para promover a WSE;
- propiciar a redução do tempo de trabalho dos engenheiros e especialistas de domínio no desenvolvimento de ontologias.

Os resultados produzidos durante o desenvolvimento deste trabalho foram:

- desenvolvimento incremental do *framework*, aplicando-o em diferentes metamodelos: primeiro em dicionário de banco de dados, depois em textos e diagramas UML;
- criação de um ambiente para composição do *framework* e análise dos resultados;
- definição de transformações necessárias entre os metamodelos das ferramentas para promover a interoperabilidade;
- definição dos requisitos necessários para as ferramentas participantes do *framework*;
- abstração da arquitetura para ontologias no domínio da Educação fundamentada no uso de metadados padrão e recursos existentes.

As ontologias geradas por processos automatizados ou semi-automatizados devem passar por refinamentos posteriores presentes na engenharia progressiva, mas este trabalho será menor do que o despendido para sua construção junto aos especialistas, desde o princípio.

1.6 Metodologia

A metodologia para atingir os objetivos propostos pode ser resumida nas etapas: estudos do contexto técnico do trabalho, estudos no domínio da Educação, criação de um ambiente experimental, construção do *framework* proposto e realização da prova de conceito.

Os estudos do contexto técnico contaram com a seguinte seqüência de pesquisas:

- Detalhamento dos principais conceitos envolvidos com WS abordando seu contexto de utilização, as diferenças entre anotação e ontologia, a arquitetura padrão proposta pelo W3C, Engenharia de Ontologias e sua abordagem na Engenharia de Software, as ferramentas disponíveis para representação das ontologias e o Aprendizado de Ontologias;
- Compreensão do padrão *Model Driven Architecture* (MDA) e *Meta Object Facility* (MOF), propostos pelo OMG, para meta-metamodelos, representação em arquivos serializados denominados *XML Metadata Interchange* (XMI) e construção de ontologias com *Ontology Definition Metamodel* (ODM).
- Enfoque para *Ontology Web Language* (OWL), proposta pelo W3C, em suas versões *OWL Lite*, *OWL Description Logics* (DL) e *OWL Full*;
- Entendimento do funcionamento de algumas ferramentas e ambientes para AO.

Já os estudos no domínio da Educação permitiram:

- Visualizar o cenário e expectativas da Web Semântica na Educação (WSE);
- Destacar os padrões de metadados existentes no domínio da Educação para vislumbrar a real utilidade de sua definição e representação em ontologias;
- Compreender os recursos presentes nos sistemas educacionais existentes, como *scripts* de criação de base de dados e metadados padrão como *IMS Learning Design*;
- Criar um projeto de aprendizagem exemplo para o curso de Computação Gráfica para experimentar o uso de metadados padrão.

A etapa criação de um ambiente experimental realizou as seguintes atividades:

- Escolha de ferramentas disponíveis para uso;
- Testes iniciais com várias ferramentas;
- Investigação dos metamodelos considerados pelas ferramentas para entrada e saída;

- Verificação da aderência aos recursos de entrada selecionados para este trabalho – modelo de dados, diagramas UML e textos.

Na etapa para Construção do *framework* proposto contou com as atividades:

- Definição dos metamodelos envolvidos e sua correspondência com OWL;
- Unificação dos conceitos em blocos no *framework* proposto;
- Definição de requisitos para as ferramentas participarem do *framework* proposto;
- Inserção de atividades do AO nas atividades iniciais da Engenharia de Ontologias.

Na Realização da Prova de Conceito as seguintes atividades foram realizadas:

- Solicitação de recursos existentes em sistemas educacionais para as instituições responsáveis como o acesso ao dicionário de dados do banco de dados do CoL⁴ - uma ferramenta gerenciadora de cursos pela Web disponível para toda a Universidade de São Paulo, desenvolvido pelo LARC⁵ (Laboratório de Arquitetura e Redes de Computadores) e os *scripts* de criação do banco de dados do sistemas Teleduc⁶ - um software livre concebido pela Universidade Estadual de Campinas (Unicamp);
- Investigação de sistemas educacionais para autoria de recursos baseados no metadado padrão IMS LD para construir o curso de Computação Gráfica do Interlab dentro do projeto TIDIA-Ae;
- Realização das atividades propostas sugeridas;
- Formalização dos resultados atingidos;
- Avaliação dos resultados do trabalho.

Os estudos contaram com pesquisas realizadas em livros, revistas e anais de congressos e participações em eventos na área. As pesquisas foram realizadas principalmente em bibliotecas digitais disponibilizadas pelos convênios firmados pela CAPES e USP com instituições como IEEE e ACM e *Sun SITE Central Europe* (CEUR). As participações no Workshop de Web Semântica (WWS 2004), no Simpósio Brasileiro de Sistemas de Informação (SBSI 2005), no *International Workshop on Model-Driven Enterprise Information Systems* (MDEIS 2008) e no *Second Brazilian Workshop on Semantic Web and Education* (WSWEd'08) contribuíram na atualização dos conceitos presentes neste trabalho.

⁴ <http://col.usp.br/portal/>

⁵ <http://www.larc.usp.br/site/index.html>

⁶ <http://teleduc.nied.unicamp.br/teleduc/>

1.7 Organização do trabalho

Este trabalho, além desta introdução, contém mais cinco capítulos, cujos conteúdos estão descritos a seguir:

O capítulo 2 – Web Semântica e Ontologias - apresenta os principais conceitos, métodos, técnicas, linguagens e padrões existentes no mundo da WS e ontologias. Destaque especial será dado ao Aprendizado de Ontologias.

O capítulo 3 – Web Semântica na Educação - contextualiza o ambiente educacional dentro da perspectiva da WS. Ressalta principalmente os conceitos de objetos de aprendizagem e projeto instrucional. Relaciona estudos com ferramentas tecnológicas utilizadas no processo de ensino-aprendizado baseado na Web.

O capítulo 4 – *Framework* Proposto - explica a constituição do *framework* com seus blocos e relacionamentos. Destaca os metamodelos e padrões necessários para representação inicial da ontologia, os requisitos para participação e as atividades a serem realizadas pelo engenheiro de ontologias.

O capítulo 5 – Prova de Conceito no Domínio da Educação - apresenta detalhes da aplicação do *framework* proposto, os resultados obtidos e discussões relevantes.

O capítulo 6 – Considerações finais - apresenta a síntese do trabalho e sugestões de alguns possíveis trabalhos futuros decorrentes da pesquisa realizada.

Os anexos A, B, C, D e E detalham respectivamente as ferramentas para Web Semântica, o *Ontology Definition Metamodel* (ODM), os métodos e técnicas em Aprendizado de Ontologias, os padrões de projeto de aprendizagem com ênfase para o *IMS Learning Design*, e um resumo da aplicação do IMS LD na construção do projeto de aprendizagem do Curso de Computação Gráfica.

2 WEB SEMÂNTICA E ONTOLOGIA

Web Semântica foi definida em 2001 e desde então apresenta inúmeras áreas de pesquisa e estudo. Este capítulo traz as principais definições envolvendo WS e Ontologia, como cenário atual, metadados e anotação semântica, definição de ontologias, arquitetura W3C, Engenharia de Ontologias e Aprendizado de Ontologias.

2.1 Cenário Atual

Tim Berners-Lee disponibilizou o primeiro site em agosto de 1991 e, a partir deste momento, milhares de páginas começaram a circular na Web, chegando nesta década ao número de 10 bilhões (SHADBOLT; HALL, 2006). Esta explosão deve-se à simplicidade de seus pilares para apresentação e navegação na rede: o protocolo HTTP (*HyperText Transfer Protocol*) e a linguagem HTML.

Porém, a grande maioria das páginas criadas permite apenas análises baseadas no nível sintático e léxico, orientadas para o entendimento humano, e limitam o tratamento por parte das máquinas (BERNERS-LEE; HENDLER; LASSILA, 2001; OBERLE et al., 2005). Para permitir a colaboração entre pessoas e máquinas, surgiu a WS. Nas palavras de Berners-Lee: “não é uma Web separada, mas uma extensão da atual, na qual a informação é utilizada com significado bem definido, não-ambíguo, integrada, aumentando a capacidade das máquinas para trabalharem em cooperação com as pessoas”.

Nesta visão, os recursos disponíveis são acessíveis não somente por seres humanos, mas também por processos automatizados. Esta automação depende da elevação do *status* da Web de “*machine-readable*” (lida automaticamente) para algo que é chamado de “*machine-understandable*” (entendida automaticamente). Neste sentido, a WS prevê tecnologias para garantir a geração de recursos inteligentes, ou seja, recursos que conhecem sobre seu conteúdo de tal forma que processos automatizados possam saber o que fazer com eles. Os domínios interessados nesta abordagem são do conhecimento científico, do jornalismo, das informações pessoais, das vendas via Web e do turismo, entre outros (UREN et al., 2006).

Alguns projetos já implementaram os conceitos da WS como o projeto POPS⁷ (*People, Organizations, Projects, Skills*) da NASA (*National Aeronautics and Space Administration*), o projeto TAO⁸ (*Transitioning Applications to Ontologies*) da Universidade de Sheffield, o projeto Flink⁹ realizado na OUNL (*Open Universiteit Nederland* ou Universidade Livre de Amsterdam), as iniciativas Confoto¹⁰ e Swoogle¹¹ (WARREN, 2006; GOTH, 2007).

2.2 Metadados e Anotação Semântica

A informação semântica necessária para WS aparece em metadados ou em anotações semânticas. O conceito de metadados é bastante simples: metadados são dados sobre dados. Os metadados podem descrever tanto um simples dado, quanto um documento, uma página da Web, ou até mesmo uma pessoa, uma coleção, um sistema, um equipamento ou uma organização; enfim, qualquer recurso. Bibliotecários são especialistas há muito tempo na produção e padronização de metadados, à medida que extraem de documentos - que seriam os dados - informações de indexação e catalogação - que seriam os metadados, para oferecer aos usuários caminhos, para que estes possam buscar os documentos de que necessitam (MILSTEAD; FELDMAN, 1999).

Metadados descrevem os recursos da Web com a finalidade de facilitar a sua descoberta, localização e utilização. Motores de busca, ao utilizarem estes metadados, proporcionam consultas mais precisas, envolvendo não somente palavras, mas propriedades descritas como: o autor do recurso, o formato do recurso, a data do recurso, palavras chave, entre outras.

Os metadados podem estar inseridos nos próprios recursos que descrevem ou serem armazenados em repositórios, na forma de anotações. Anotação semântica identifica, formalmente, conceitos e relações entre conceitos nos recursos Web para serem utilizados por máquinas para recuperação de informação e para garantia de interoperabilidade (UREN et al., 2006). Anotador e consumidor da anotação devem

⁷ <http://xtech06.usefulinc.com/schedule/paper/147>

⁸ <http://www.tao-project.eu/>

⁹ <http://flink.semanticWeb.org/>

¹⁰ <http://www.confoto.org/home>

¹¹ <http://swoogle.umbc.edu/>

compartilhar o mesmo significado. Para isto, é providenciada referência a uma ontologia.

A WS propõe anotar o conteúdo dos documentos usando informações semânticas de um domínio ontológico. O resultado são páginas interpretadas automaticamente através de agentes ou serviços Web semânticos (OBERLE et al., 2005; FENSEL, 2003).

Para WS, esquemas de metadados formam uma base para proporcionar um entendimento semântico comum. Os primeiros formatos projetados para atingir este objetivo foram o *Resource Description Framework* (RDF) do W3C (W3C, 2004) e *Topic Maps* do *International Standardization Organization* (ISO, 1999). Os dois formatos facilitam a descrição de recursos e estabelecem relacionamentos entre eles. A semântica marcada e providenciada pelo RDF e *Topic Maps* pode ser estendida pela referência a um conhecimento externo como vocabulários controlados, taxonomias e ontologias.

O estado da arte no domínio de anotação semântica pode ser dividido em múltiplas categorias, como a plataforma de anotação semântica (usando representação formal) *versus* uma não-semântica (usando representação não formal), manual *versus* automática, *frameworks versus* ferramentas, plataformas genéricas *versus* plataformas específicas (ZOUAQ; NKAMBOU; FRASSON, 2007). A noção de metadados expandiu-se, tomando novas formas de ser gerido. Por exemplo, geração automática de metadados, onde o metadado é gerado como parte do processo criativo – autoria do recurso -, ou de parte do contexto de uso dos recursos - as informações sobre os usuários e suas ações nos sistemas são capturadas automaticamente -, ou inferida a partir do conteúdo de um recurso. Assim os metadados não são utilizados somente para fins descritivos tradicionais, mas sim para exprimir relações contextuais e de apoio aos sistemas adaptativos (NILSSON, 2008).

Uren et al. (2006) classificaram as ferramentas para anotação em manual, automática (sugestões de anotações, mas que ainda requerem a intervenção de trabalhadores do conhecimento e ferramentas para adquirir anotações, automaticamente, em grande escala) e integrada (para criação e anotação de recursos simultaneamente), enumeradas no Anexo A.

Existem também *frameworks* para anotação como o Annotea¹² e o CREAM (*CREating Metadata for the Semantic Web*).

O *framework* Annotea é um projeto do W3C para especificação da infra-estrutura necessária para anotação de documentos Web com ênfase no uso colaborativo de anotações. Providencia o XPointer, um método para localizar anotações num documento através da identificação de fragmentos de recursos URI (*Uniform Resource Identifier*). Tem um estilo semi-formal de anotação no qual anotações são comandos em texto livre sobre documentos. Tem sido instanciado nas ferramentas como Amaya, Annozilla e Vannotea.

Já o *framework* CREAM é um projeto da Universidade de Karlsruhe e possui módulos para suporte automático para anotadores, para gerenciamento de documentos e para inferência nas anotações (HANDSCHUH; STAAB, 2003). Exemplos de ferramentas baseadas neste *framework* são S-CREAM e M-OntoMat-Annotizer.

Diante de tantas ferramentas para anotação, Uren et al. (2006) identificaram requisitos técnicos necessários para os sistemas de anotação semântica:

- ☐ possuir formato padrão (mecanismos de aderência entre recursos heterogêneos);
- ☐ ser colaborativo (anotação presente no contexto diário de trabalho dos produtores de recursos);
- ☐ prover colaboração (colaboração entre diferentes usuários de diferentes domínios para reuso dos documentos inteligentes);
- ☐ tratar ontologia (união e declaração explícita de múltiplas ontologias garantindo consistência entre elas);
- ☐ tratar formatos de documentos heterogêneos (documentos em formatos diversos além do HTML e XML (*eXtensible Markup Language*) como arquivos de editores de texto, planilhas eletrônicas, gráficos, entre outros);
- ☐ acompanhar a evolução do documento (as mudanças dos documentos refletidas na sua anotação);
- ☐ armazenar as anotações (o modelo da WS separa o conteúdo de sua semântica);
- ☐ automatizar a anotação (facilidade para adquirir conhecimento embutido numa coleção de documentos).

¹² <http://www.w3.org/2001/Annotea/>

Recursos e anotações semânticas devem ser produzidos e mantidos com ferramentas integradas de tal forma que suas informações permaneçam íntegras desde a criação do recurso até sua situação atual.

2.3 Ontologias

Metadados e ontologias são complementares. Pesquisas em ontologias têm origem na filosofia com a natureza e organização das “coisas” e onde ontologia significa uma explanação sistemática da existência do ser. Segundo Gruber (1995) e Studer, Benjamins e Fensel (1998), ontologia é uma especificação formal explícita compartilhada de uma conceitualização. Formal refere-se ao fato que uma ontologia precisa ser lida automaticamente. Explícita significa que o tipo de conceitos usados e as restrições no seu uso são explicitamente definidos. Compartilhada significa que uma ontologia captura um conhecimento consensual, isto é, não é para um indivíduo, mas sim, aceito por um grupo. Conceitualização refere-se a um modelo abstrato de algum fenômeno no mundo pela identificação de conceitos relevantes deste fenômeno. Guarino (1997) também destaca o compromisso ontológico para uma conceitualização particular do mundo.

Em Ciência da Computação, o termo ontologia refere-se a um artefato de engenharia, constituído por um vocabulário específico para descrever um modelo particular do mundo, adicionando um conjunto explícito de suposições ou decisões, relacionando o significado das palavras no vocabulário (MAEDCHE, 2002). Ou, em outras palavras, uma ontologia ajuda a prover uma descrição formal explícita de conceitos em um domínio de discurso, propriedades de cada conceito descrevendo várias funcionalidades e atributos de conceitos, e restrições das propriedades (BERGAMASCHI et al., 2005).

Formalmente pode-se definir uma ontologia como uma tupla (C, I, R, F, A) , onde:

- **C** é um conjunto não vazio de conceitos - abstrações usadas para descrever os objetos do mundo;
- **I** é um conjunto de indivíduos e suas relações - os objetos atuais do mundo, também conhecidos como instâncias dos conceitos, *facts* ou *claims*;
- **R** é o conjunto de relações definidas em C; representam as interações entre os conceitos do domínio, independentes de sua utilização e também conhecido

como conjunto de ações e heurísticas. R é particionado em dois sub-conjuntos H e N . H é um conjunto de todas as afirmações na qual a relação é uma relação taxonômica (expressa uma hierarquia) e N é um conjunto de todas afirmações não-taxonômicas (expressa uma ação);

- F é o conjunto de funções definidas em C e seu retorno é um conceito;
- A é o conjunto de axiomas - predicados da lógica de primeira ordem que restringem o significado dos conceitos, relações e funções; o resultado é sempre uma sentença verdadeira.

Complementar à esta tupla, existem os conceitos para propriedades, taxonomia e conceito raiz (*Top*). As propriedades são características para descrever e especificar os conceitos. A taxonomia (de conceitos e de relações) é utilizada para organizar o conhecimento em diferentes maneiras de relacionamento (generalização, especialização, composição, hierarquia de subconjuntos), acordando para as restrições envolvidas em múltiplos relacionamentos taxonômicos. Implica um conhecimento estendido como: se A é uma subclasse ou sub-relação de B , então cada instância de A precisa também ser uma instância de B . Raiz é o conceito de nível mais alto na hierarquia.

Diferentes esforços têm proposto uma classificação para ontologias. O Quadro 1 mostra, no tempo, os autores e suas diferentes abordagens.

Autores	Classificação	Tipos	Breve Descrição
(MIZOGUCHI; TIJERINO; IKEDA, 1995)	Quanto ao conteúdo	Ontologias de domínio	Reutilizáveis no domínio, fornecem vocabulário sobre conceitos bem como seus relacionamentos, atividades e regras que os governam
		Ontologias de tarefa	Fornecem um vocabulário sistematizado de termos, especificando tarefas que podem ou não estarem no mesmo domínio
		Ontologias gerais	Incluem um vocabulário relacionado a coisas, eventos, tempo, espaço, casualidade, comportamento, funções etc
(USCHOLD; GRUNNINGER, 1996)	Quanto ao grau de formalismo	Ontologias altamente informais	São expressas livremente em linguagem natural
		Ontologias semi-informais	São expressas em linguagem natural, de forma restrita e estruturada
		Ontologias semiformais	São expressas numa linguagem artificial definida formalmente
		Ontologia rigorosamente formal	Os conceitos são definidos com semântica formal, teoremas e provas
(VAN-HEIJST; SCHREIBER; WIELINGA, 1997)	Quanto ao tipo de estrutura	Ontologias terminológicas	Especificam termos usados para representar o conhecimento em um domínio (por exemplo, os léxicos)
		Ontologias de informação	Especificam a estrutura de registros de banco de dados (por exemplo, os esquemas de banco de dados)
		Ontologias de modelagem de conhecimento	Especificam conceitualizações do conhecimento; têm uma estrutura interna semanticamente rica e são refinadas para uso no domínio do conhecimento que descrevem
	Quanto à conceitualização	Ontologias de aplicação	Contêm as definições necessárias para modelar o conhecimento em uma aplicação
		Ontologias de domínio	Expressam conceitualizações que são específicas para um determinado domínio de conhecimento
		Ontologias genéricas	Similares às ontologias de domínio, mas os conceitos que as definem são considerados genéricos e comuns a vários domínios
		Ontologias de representação	Explicam as conceitualizações que estão por trás dos formalismos de representação do conhecimento
(JASPER; USCHOLD, 1999)	Quanto à aplicação	Ontologias de autoria neutra	É escrita em uma língua única e depois convertida para uso em diversos sistemas, reutilizando-se as informações
		Ontologias como especificação	Cria-se uma ontologia para um domínio, a qual é usada para documentação e manutenção no desenvolvimento de software
		Ontologias de acesso comum à informação	Torna a informação inteligível, proporcionando conhecimento compartilhado dos termos
(GÓMEZ-PÉREZ; BENJAMINS, 1999)	Quanto ao grau de genericidade	Ontologias de representação	Definem as primitivas de representação - como frames, axiomas, atributos e outros - de forma declarativa
		Ontologias gerais	Trazem definições abstratas, necessárias para a compreensão de aspectos do mundo, como tempo, processos, papéis, espaço, seres, coisas, etc
		Ontologias centrais	Definem os ramos de estudo de uma área e/ou conceitos mais genéricos e abstratos desta área
		Ontologias de domínio	Tratam de um domínio mais específico de uma área genérica de conhecimento, como direito tributário, microbiologia, etc
		Ontologias de aplicação	Procuram solucionar um problema específico de um domínio, como identificar doenças do coração, a partir de uma ontologia de domínio de cardiologia
(HAAV; LUBI, 2001)	Quanto à estrutura	Ontologias de alto nível	Descrivem conceitos gerais relacionados a todos os elementos da ontologia os quais são independentes do problema ou domínio
		Ontologias de domínio	Descrivem o vocabulário relacionado a um domínio, como medicina e automóveis
		Ontologias de tarefa	Descrivem uma tarefa ou atividade, como, diagnóstico ou compras
(MAEDCHE, 2002)	Quanto ao grau de genericidade	Ontologias de alto-nível	Descrivem conceitos muito gerais como espaço, tempo, evento, etc.
		Ontologias de domínio	Descrivem o vocabulário relacionado a um domínio genérico, através da especialização de conceitos introduzidos nas ontologias de alto-nível
		Ontologias de tarefa	Descrivem um vocabulário relacionado a uma tarefa ou atividade genérica, através da especialização de conceitos introduzidos nas ontologias de alto-nível
		Ontologias de aplicação	São as ontologias mais específicas por serem utilizadas dentro das aplicações

Quadro 1 - Tipos de Ontologias, adaptado de (ALMEIDA, 2003)

Mesmo sem um consenso entre os autores na classificação, observa-se que os tipos de ontologias apresentados guardam semelhanças entre suas descrições, principalmente, para ontologias de domínio, de aplicação e de tarefa.

Existem muitas razões para se desenvolver ontologias (BERGAMASCHI et al., 2005), como para compartilhar um entendimento comum da estrutura da informação entre pessoas ou entre agentes de software, para garantir reuso do conhecimento do domínio, para fazer uma definição explícita do domínio, para ajudar novos usuários a aprender o significado dos termos no domínio ou para facilitar sua busca, para separar o domínio de conhecimento do conhecimento operacional e/ou para analisar o domínio de conhecimento.

Ontologias podem ser usadas numa variedade de sistemas de informação, como por exemplo, nos sistemas para gerenciamento de conteúdo para promover a independência entre o conteúdo e sua apresentação, nos sistemas P2P¹³ (*peer-to-peer*) para compartilhar módulos que capturam conhecimento na rede para melhorar a cooperação, nos sistemas multi-agentes, providenciando um vocabulário comum para os agentes, nos *Web Services* (integradas com os registros de serviços) para fornecer informações semânticas para serviços Web e descrevê-los de maneira inteligível e nos sistemas de recuperação de informações (ZAVITSANOS; VOUROS; PALIOURAS, 2006; SHADBOLT, 2008).

A ontologia, presente nos sistemas de informação, providencia comunicação e cooperação eficiente entre pessoas de uma organização, interoperabilidade entre sistemas e outros benefícios na engenharia de sistemas como reusabilidade e integridade.

Como as ontologias são teorias formais que descrevem um domínio, elas requerem uma linguagem lógica para representá-las. Como as linguagens têm diferentes níveis de formalização, também apresentam diferente poder de expressividade. As linguagens para expressar uma ontologia precisam apresentar algumas características:

- ☐ expressar o domínio de conhecimento ao qual se aplica, isto é, as principais informações estáticas e os elementos conhecidos sobre o domínio;
- ☐ prover algum mecanismo de inferência, isto é, como as estruturas estáticas representadas no domínio são transportadas para um mecanismo de raciocínio;
- ☐ incluir mecanismos para o reuso e para a integração com ontologias previamente desenvolvidas, providenciando mecanismos para mapeamento de conceitos

¹³ Tecnologia para estabelecer uma espécie de rede de computadores virtual, onde cada estação possui capacidades e responsabilidades equivalentes.

similares entre diferentes ontologias, permitindo a importação de outras ontologias, e reconciliando inconsistências;

- ter um apoio gráfico (para leitura humana);
- ser multilinguagem, ou seja, mesmos conceitos podem ser descritos em diferentes linguagens (para leitura humana);
- realizar a operação de correspondência (*match*) identificando conceitos similares em diferentes ontologias. Resolver conflitos gerados:
 - conflitos de representação, isto é, dois modelos descrevem o mesmo conceito de maneiras diferentes;
 - conflitos fundamentais, isto é, violação de meta-metamodelos (como um tipo de restrição, quando o mesmo elemento é descrito pelo significado de tipos diferentes de dados em diferentes ontologias, conflitos de cardinalidade, ...)

Nos últimos anos surgiram várias linguagens para representação ou construção de ontologias, algumas relacionadas especificamente para WS, outras não. Algumas linguagens existentes criaram taxonomias e relações entre múltiplas classes; outras formaram definições mais complexas, usando axiomas lógicos. Muitos estudos comparam as diferenças sintáticas, semânticas e de utilização entre as linguagens como Gómez-Pérez e Corcho (2002), Pulido et al. (2006) e Bergamaschi et al. (2005).

A escolha de uma linguagem ou outra se dará pelo seu poder de expressividade, existência de ferramentas para sua edição e integração da ontologia com ela criada, familiaridade com a representação do formalismo, necessidade de mecanismos de raciocínio da aplicação e pela existência de ferramentas para anotação dos recursos, entre outros fatores.

Existem ainda as linguagens especializadas em consultas e recuperação das ontologias existentes. No Anexo A encontra-se as principais linguagens para representação e para consultas de ontologias com a referência correspondente.

2.4 Arquitetura

A cooperação esperada na WS pode ser realizada com padronização de protocolos e linguagens para compartilhar e reutilizar informações (ZAVITSANOS; VOUROS; PALIOURAS, 2006). O consórcio W3C coordena a WS apoiado pela indústria de

software (W3C, 2007). Este consórcio propôs uma arquitetura em camadas para construir aplicações que envolvam WS¹⁴ que está na sua quarta versão, apresentada na Figura 2. Esta arquitetura define as tecnologias necessárias para que os conteúdos dos recursos Web possam ser compreendidos pelas máquinas e sua contribuição é prover um conjunto de padrões - linguagens, sintaxe comum, métodos, entre outros.

Figura 2 – Arquitetura da Web Semântica, adaptada de (W3C, 2007)

Na primeira camada da arquitetura (1), o URI ou IRI (*Internationalized Resource Identifiers*), fornece um mecanismo de identificação único, ou endereço global, para os recursos disponibilizados na Web, permitindo que textos e imagens possam ser lidos pelos computadores em qualquer lugar, via HTTP.

Na segunda camada (2), a XML permite estruturar os documentos da Web numa hierarquia de árvore, baseada em marcas, criadas pelo usuário. O papel da XML é limitado à sintaxe, para portar a troca de dados.

A terceira camada, Intercâmbio de Dados (3a), realizada com o RDF, dá significado aos dados com o modelo de dados e o RDF-S (3b) propicia a representação do conhecimento através de expressões lógicas. O RDF pode ser utilizado para afirmativas simples sobre os recursos Web ou outra entidade nomeada. Uma

¹⁴ <http://www.w3.org/2007/03/layerCake.png>

afirmação simples é uma declaração sobre uma propriedade com um valor particular de uma entidade. RDFS estende o RDF com hierarquia de classes e propriedades, habilitando a criação de ontologias simples. Os metadados dos recursos são representados em RDF (3a) e definem formalmente, tanto as instâncias, como os conceitos da ontologia. Relacionamentos entre os conceitos são descritos com a semântica formal fornecida pela Ontologia (4a). A ontologia constrói um vocabulário compartilhado para modelar um determinado domínio com suas propriedades e relações e permite que as máquinas pesquisem e tomem decisões a respeito do significado dos dados e possam inferir novos fatos a partir do conhecimento recém adquirido. A ontologia pode ser expressa com a OWL, em alguma de suas três versões: *OWL Lite*, *OWL DL* (*OWL Description Logic*) e *OWL Full*.

A versão *OWL Lite* garante implementações simples e eficientes como hierarquias de classificação e restrições simples. Por exemplo, a cardinalidade máxima ou mínima assume apenas os valores 0 ou 1. Propriedades podem ser transitivas, simétricas, funcionais e funcionais inversas.

OWL DL garante a expressividade da lógica de descrições. Classes podem ser construídas por união, interseção e complemento, pela enumeração de instâncias e podem ter disjunções. Tipos de dados são mantidos cuidadosamente separados, por exemplo, uma classe não pode ser instância e propriedade ao mesmo tempo.

OWL Full tem a expressividade da *OWL DL* e a possibilidade de uso de novas metaclasses (classes de classes) definidas em RDFS. Por isso, não há garantia de eficiência computacional e não cabem as restrições de separação de tipos da versão anterior.

As linguagens menos expressivas (*OWL Lite* e *OWL DL*) estão contidas dentro das mais expressivas (*OWL DL* e *OWL Full*), de maneira que uma ontologia definida numa linguagem menos expressiva é aceita por uma linguagem mais expressiva; a recíproca, porém, não é verdadeira.

A função das camadas 3b e 4a é a definição de uma ontologia ou formalismo de metadados.

Ainda na quarta camada, Regras (4b), providencia uma linguagem interoperacional para descrever o conjunto de deduções aplicáveis à coleção de dados - como, dada uma base de informações representada numa ontologia, novas informações derivarem dos dados existentes. O formato padrão usa o RIF (*Rule Interchange Format*) que permite compartilhar regras lógicas criadas em diferentes linguagens,

permitindo o intercâmbio delas entre diversos sistemas baseados em regras na Web. RuleML¹⁵ (*Rule Modeling Language*) e SWRL são exemplos de linguagens que implementam esta camada. SWRL garante cláusulas de Horn para serem usadas numa base de conhecimento em OWL. Regras SWRL podem, então, tirar proveito de classes OWL, instâncias e propriedades dos seus antecedentes e conseqüentes. Na camada Consulta (4c) tem-se a SPARQL¹⁶, uma linguagem de consulta similar ao SQL, que permite a unificação das coleções descentralizadas de dados em RDF e faz requisições de busca com conjunções e disjunções. Os mecanismos de procura podem inferir conceitos associados, a partir das propriedades, como ocorre com as propriedades simétricas.

A quinta camada, Lógica Unificada (5), define um modelo semântico singular, teórico e comum que une os formalismos como XML, RDF, RDFS, SPARQL e OWL com o objetivo de especificar linguagens de lógica para facilitar a construção de inferências sobre instâncias e recursos.

A sexta camada, Prova (6), é a especificação de uma linguagem para provar que as informações trocadas entre agentes e máquinas são verdadeiras. A linguagem de prova providencia uma forma para descrever os passos seguidos até a conclusão dos fatos. Todas as provas podem, então, serem passadas para verificação e providenciam caminhos mais curtos para criação de novos fatos no sistema. Isto tira a necessidade de percorrer cada nó toda vez que se faz uma mesma dedução. Nesta camada executam-se as regras de inferência da quinta camada, onde os agentes têm mais poder para realizar os mecanismos de raciocínio sobre conceitos e relacioná-los na camada de Ontologia, e prova-se que as informações trocadas são verdadeiras. Nesta camada verifica-se a consistência dos dados acessados na WS. Um exemplo é a máquina de inferência Bossam que apoia subconjuntos de regras SWRL e RuleML.

As camadas Confiança (7) e Criptografia (9) introduzem os conceitos de segurança na WS e avaliam, se a prova está correta ou não, visando garantir a procedência de um documento, o que é fundamental para decidir se a informação é confiável ou não. Trata-se de uma característica importante, na qual blocos de dados criptografados são utilizados para garantir a autenticidade das fontes e a confiabilidade das informações consultadas.

¹⁵ <http://www.ruleml.org/>

¹⁶ <http://www.w3.org/TR/rdf-sparql-query/>

A oitava camada, Aplicações e Interfaces com Usuário (8), possibilita a troca de dados, informações ou conhecimento entre aplicações Web.

Na visão da WS todas camadas estão num ambiente no qual pode-se encontrar dados verdadeiros, realizar deduções e aplicar regras. O objetivo é melhorar a vida dos usuários pela agregação e criação de informações novas e verdadeiras na Web.

2.5 Engenharia de Ontologias

Engenharia de Ontologias (EO) é formalmente definida como um conjunto de atividades importantes para o processo de desenvolvimento de ontologias e para o ciclo de vida da ontologia, bem como o de todas as ferramentas e linguagens envolvidas (SIMPERL; TEMPICH, 2006; PINTO; TEMPICH; SURE, 2004; MAEDCHE, 2002). Assim EO tem atividades incluídas nos processos de construção, projeto, avaliação, validação, manutenção, integração, compartilhamento e reutilização de ontologias.

A WS depende da interação complexa de muitas tecnologias envolvendo ontologias (OBERLE et al., 2005). O resultado teórico obtido na EO nos últimos quinze anos é de valor incontestável para a perspectiva de uso em alta escala das tecnologias semânticas.

Dentre elas destacam-se a metodologia, a abordagem da Engenharia de Software e as principais ferramentas utilizadas.

2.5.1 Metodologia

A concepção de ontologias deve ser conduzida como qualquer outro projeto de software, no sentido de serem tomadas decisões de projeto que determinam sua qualidade, baseada em critérios como eficiência, legibilidade, portabilidade, extensibilidade, interoperabilidade e reuso (USCHOLD; GRUNINGER, 1996; GUARINO, 1997; FERNÁNDEZ-LÓPEZ, 1999; STAAB; MAEDCHE, 2000).

Segundo (MARIETTO et. al, 2002), o desenvolvimento e o uso de metodologias para a construção de ontologias é fundamental, na medida que retiram o caráter subjetivo desta atividade.

Metodologias dividem o processo de construção da ontologia em vários estágios e propõem várias atividades para cada um. Os estágios geralmente aceitos para construir uma ontologia são especificação, conceitualização, formalização, implementação e manutenção (PINTO; TEMPICH; SURE, 2004).

A Figura 3 apresenta as atividades divididas em gerenciamento, desenvolvimento, suporte e pós-desenvolvimento ontológico (SIMPERL; TEMPICH, 2006; GÓMEZ-PÉREZ, 2007). No Gerenciamento Ontológico são elaborados cronogramas, controles, garantia de qualidade e o Estudo de Viabilidade. Análise do Domínio, Conceitualização e Implementação são as atividades consideradas clássicas do desenvolvimento. As atividades de Suporte – Reuso de Ontologias, Aquisição de Conhecimento, Avaliação, e Documentação –, são realizadas em paralelo com as atividades principais do Desenvolvimento. A Manutenção e o Uso de ontologia são atividades consideradas Pós-desenvolvimento.

Figura 3 – Atividades da Engenharia de Ontologias, adaptada de (SIMPERL; TEMPICH, 2006)

No Estudo de Viabilidade é realizada a especificação para identificar o objetivo e o escopo da ontologia. O objetivo responde a questão: “Por que a ontologia será construída?”. O escopo responde a questão “Qual é a intenção de uso e quem são os usuários finais?”. O Estudo de Viabilidade examina se uma aplicação baseada na

ontologia, em um dado contexto, é a maneira certa para resolver o problema em questão.

Na Análise de Domínio, a equipe de engenharia – especialistas de domínio e engenheiros de ontologia – realiza uma análise profunda dos pré-requisitos do projeto como cenários, questões de competência e soluções existentes. O resultado é apresentado num documento que, além dos requisitos da ontologia, descreve o domínio modelado com informações sobre seus casos de uso, tamanho estimado, fontes utilizadas, participantes do processo de construção e metodologia utilizada.

A Conceitualização descreve, em um modelo conceitual, a ontologia a ser construída, e reúne a especificação presente na Análise de Domínio. O modelo conceitual consiste de conceitos do domínio e seus relacionamentos.

Na Implementação, o modelo conceitual é formalizado numa linguagem de representação do conhecimento. Conceitos são definidos com axiomas para restringir a possibilidade de interpretação do seu significado, relações taxonômicas do tipo *is-a* (classe e superclasse) ou *part-of* e relações não-taxonômicas. Já a Manutenção garante refletir corretamente as alterações e implementações na ontologia já construída.

A Aquisição de Conhecimento obtém instâncias do domínio usando técnicas de elucidação em domínios especialistas ou pela referência à bibliografia relevante. A Avaliação julga tecnicamente a qualidade da ontologia. A Documentação relata o que foi feito, como foi feito e por que foi feito.

Jimenez-Ruiz e Berlanga (2006) estudaram diferentes metodologias e classificaram-nas em dois grupos. O primeiro grupo representa as metodologias clássicas com a abordagem no desenvolvimento centralizado da ontologia conforme exemplos presentes no Quadro 2. Embora estas metodologias descrevam muitas fases para o desenvolvimento de ontologias, elas negligenciam ações para colaboração na construção da ontologia.

Referência	Identificação	Breve Descrição
(LENAT; GUHA, 1990)	Cyc	Codifica manualmente o conhecimento implícito das diferentes fontes, e, quando já se tem conhecimento suficiente na ontologia, um novo consenso pode ser obtido por ferramentas que utilizam linguagem natural
(BERNARAS; LARESGOITI; CORERA, 1996)	KACTUS	Método recursivo que consiste em uma proposta inicial para uma base de conhecimento; quando é necessária uma nova base em domínio similar, generaliza-se a primeira base em uma ontologia adaptada a ambas aplicações; quanto mais aplicações, mais genérica a ontologia fica. Esta proposta de desenvolvimento de ontologias está condicionada à aplicação.
(USCHOLD; GRUNNIGER, 1996)	Projeto <i>Enterprise Ontology</i>	Identifica o propósito, os conceitos e relacionamentos entre os conceitos, além dos termos utilizados para codificar a ontologia e, em seguida, documentá-la
(SWARTOUT; GIL, 1996)	Sensus	Constrói ontologias a partir de outras ontologias, identificando os termos relevantes para o domínio e ligando-os à ontologia mais abrangente; um algoritmo monta a estrutura hierárquica do domínio. Esta metodologia utiliza processamento de linguagem natural.
(FERNÁNDEZ-LÓPEZ, 1999)	METHONTOLOGY	Constrói uma ontologia por reengenharia sobre outra, utilizando-se o conhecimento do domínio; as atividades principais são: especificação, conceitualização, formalização, implementação e manutenção. É suportada pelo ODE (<i>Ontology Development Process</i>)
(SCHREIBER; CRUBEZY; MUSEN, 2000)	<i>CommonKADS</i>	É uma referência no desenvolvimento de sistemas baseados em conhecimento. Principal modelo construído é o Modelo de Experiência: nível do domínio (conhecimento declarativo sobre o domínio), nível de inferência (uma biblioteca de estruturas genéricas de inferência) e nível de tarefa (ordem das inferências).
(STAAB et al., 2001)	OTK (On-To-Knowledge)	Auxilia a administração de conceitos em organizações, identificando metas para as ferramentas de gestão de conhecimento e utilizando cenários e contribuições dos provedores / clientes de informação da organização
(DE NICOLA; MISSIKOFF; NAVIGLI, 2005)	UPON (<i>Unified Process for Ontology Building</i>)	Uma metodologia derivada do Processo Unificado para Desenvolvimento de Software e UML, por isso, é baseado em casos de uso, interativo e incremental e na arquitetura

Quadro 2 - Metodologias com Abordagem no Desenvolvimento Centralizado

O segundo grupo contém Co4, (KA)2, Diligent e HCOME (*Human Centered Ontology Engineering Methodology*), apresentadas no Quadro 3. Este grupo não propõe uma metodologia completa, mas soluções para construir uma ontologia em colaboração com outros engenheiros de ontologia, presentes em distintos ambientes e domínios.

Referência	Identificação	Breve Descrição
(EUZENAT, 1996)	Co4	Permite discussão sobre conhecimento introduzido em bases de conhecimento compartilhadas que correspondem a ontologias, já que deve haver um consenso sobre o conhecimento ali representado. Quando uma mudança é proposta, os usuários são notificados e podem aceitá-la ou não.
(BENJAMINS et al., 1999)	(KA)2	Modela formas de aquisição do conhecimento usando ontologias desenvolvidas em conjunto por pessoas em diferentes locais, mas que utilizam o mesmo padrão; a comunicação e coordenação são feitas via agentes inteligentes
(KOTIS; VOUIROS, 2006)	HCOME	Prevê uma participação ativa dos trabalhadores do conhecimento no ciclo de vida da ontologia
(CASANOVAS et al., 2007)	Diligent	Desenvolvimento descentralizado, facilita a colaboração e distribuição das ontologias contruídas através de um framework que acompanha todas as discussões relevantes para conceitualização

Quadro 3 - Metodologias com Abordagem no Desenvolvimento Colaborativo

Outros sistemas para desenvolvimento de ontologias existentes na literatura são Ontolingua (FARQUHAR; FIKES; RICE, 1997), WebOnto (DOMINGUE, 1998),

WebODE (ARPÍREZ et al., 2001) e Wiki@nt (BAO; HONAVAR, 2004). Embora não sejam consideradas metodologias por estarem atreladas somente ao projeto que participam, providenciam um bom *framework* para colaboração (JIMENEZ-RUIZ; BERLANGA, 2006).

Kim e Choi (2007) acrescentam outras metodologias como Mikrokosmos¹⁷ e ONIONS¹⁸ (*ONtological Integration Of Naive Sources*) e introduzem os conceitos do CMM-i (*Capability Maturity Model integration*) na Engenharia de Ontologias.

Estudos comparativos e análises entre diversas metodologias são encontrados em Kotis e Vouros (2006), Pinto, Tempich e Sure (2004), Corcho e Gómez-Pérez (2005) e Cristani e Cuel (2004). Estes estudos separam as metodologias em três gerações conforme Quadro 4.

Geração	Características Básicas	Exemplos
Primeira	Iniciam com a compreensão de como ontologias são construídas Não seguem ciclo em cascata ou iterativo Falta estágio de manutenção	TOVE (<i>TOronto Virtual Enterprise</i>)
Segunda	Influência das metodologias da Engenharia de Software	Methontology
Terceira	Distribuição com ênfase nas ações de colaboração e argumentação; evolução, gerenciamento de versões e fusão de ontologias.	HCOME Diligent

Quadro 4 - Gerações de metodologias

Existem ainda metodologias para avaliação, para integração e aprendizado de ontologias. Avaliar, em geral, significa julgar tecnicamente as funcionalidades de um produto. Avaliação refere-se à usabilidade e utilidade do produto na empresa ou em um domínio específico. Avaliação de Ontologia consiste em verificar, dada uma ontologia construída sob um ponto de vista para uma aplicação com critérios particulares, se ela atende esses propósitos particulares. Pode ser realizada comparando-se com um padrão, com diferentes fontes de dados ou com conhecimentos humanos.

Gómez-Pérez (2007) cita a existência de 55 atividades definidas no glossário de atividades do Projeto Neon¹⁹ para construção de ontologias de rede. As atividades são classificadas como opcionais ou obrigatórias, de acordo com o contexto

¹⁷ <http://crl.nmsu.edu/Research/Projects/mikro/index.html>

¹⁸ <http://www.loa-cnr.it/Ontologies.html>

¹⁹ <http://www.neon-project.org/Web-content/>

existente – buscar ontologias existentes, realizar engenharia reversa, fazer junção de ontologias existentes, entre outras - e também considera na seleção das atividades o número de ontologias já construídas pelos engenheiros de ontologias.

Para Fernández-López (1999), o que diferencia uma área de estudo na fase de “infância” para outra, em fase “adulta”, é quando suas metodologias são aceitas e utilizadas. Por exemplo, pode-se dizer que a Engenharia de Software está na fase adulta, pois suas metodologias são largamente difundidas e utilizadas. Com esta observação pode-se considerar as metodologias para ontologias ainda na infância e, embora existam muitas metodologias, nenhuma ultrapassou amplamente os limites do projeto que a criou.

2.5.2 Abordagem na Engenharia de Software

Sheth, Ramakrishnan e Thomas (2005) classificaram a semântica como: semântica implícita, semântica formal e semântica total.

Técnicas de recuperação de informação, extração de informação e lingüística computacional trabalham primariamente sobre análise de textos desestruturados, adicionando-os em repositórios de documentos com uma definição mais livre e estrutura menos formal. A semântica está implícita na origem dos recursos. Algoritmos de mineração e aprendizado podem ser aplicados nestes recursos para extrair conhecimento estruturado ou enriquecer representações estruturadas formais. Mas, uma semântica formal intrínseca não existe.

Nos campos de representação do conhecimento, inteligência artificial e gerenciamento de dados, a representação dos dados tem uma forma mais rígida e formal. Estruturas bem definidas sintaticamente são usadas para representar informações ou conhecimento e têm uma interpretação semântica definida associada a elas. As regras sintáticas definidas garantem a maneira pela qual as estruturas podem ser combinadas para representar o significado de estruturas sintáticas complexas. Em outras palavras, estas técnicas garantem uma semântica formal, que geralmente aparece na forma de ontologia. Mecanismos de representação com semântica formal proporcionam para as aplicações técnicas de raciocínio automático, garantindo maior inteligência às aplicações. Usualmente,

esforços relacionados com semântica formal envolvem limites de expressividade para poder aceitar os limites computacionais.

Por outro lado, a representação do conhecimento e o modelo relacional de dados não apresentam formas para representar e utilizar conhecimento impreciso, incerto, parcialmente verdadeiro ou aproximado. Representação e utilização desse tipo de conhecimento mais poderoso são críticos para o sucesso da WS. Esta é a semântica total. Na forma de mecanismos *fuzzy* ou probabilístico, a representação do conhecimento pode interpretar os dados com grau de incerteza e grau de participação.

Considerando a representação da semântica formal, as abordagens para modelagem, ontologias e MDA iniciaram seu desenvolvimento em paralelo como já citado anteriormente. MDA é uma das principais iniciativas do *Model-Driven Engineering* (MDE) – uma abordagem para o desenvolvimento de software que separa a especificação funcional do sistema da sua implementação numa plataforma particular. Desta maneira, modelos e conceitos podem ser compartilhados entre métodos e ferramentas diferentes (SCHIMT, 2006).

MDA é um *framework* global para interoperabilidade entre sistemas, proposta pelo OMG e define três níveis de abstração para representar diferentes pontos de vista de um sistema: *Computation Independent Model* (CIM₁), *Platform Independent Model* (PIM) e *Platform Specific Model* (PSM) (OMG, 2007b). CIM₁ mostra o sistema do ponto de vista do negócio, não se preocupa com a estrutura do sistema, é também conhecido como modelo de domínio e é criado por especialistas do domínio. PIM é o modelo dependente da computação; não apresenta detalhes de nenhuma plataforma específica, é geralmente expresso com diagramas da UML, como uma máquina virtual com tecnologia neutra. PSM é a especificação completa das características do ambiente computacional no qual o sistema será implementado. O objetivo é mover os maiores esforços na construção do CIM₁ e do PIM e gerar, automaticamente, tantos PSMs quantos ambientes computacionais forem necessários. Juntamente com a MDA existe a *Architecture-Driven Modernization* (ADM) – uma especificação proposta para detalhar o processo de desenvolvimento de artefatos de software existentes na engenharia progressiva com as ferramentas da engenharia reversa (OMG, 2008).

MDA é baseada na metamodelagem em quatro camadas do MOF, e muitos padrões complementares do OMG (OMG, 2007b), já apresentados anteriormente. As

camadas são meta-metamodelo (M3), metamodelo (M2), modelo (M1) e instância (M0).

No topo desta arquitetura está o meta-metamodelo (M3), que define uma linguagem abstrata e um *framework* para especificação, construção e gerenciamento de metamodelos neutros de tecnologia. Ele é o alicerce para definição de uma linguagem de modelagem. MOF também define um *framework* para implementação de repositórios para armazenamento de metadados, como os modelos construídos para os sistemas descritos por metamodelos. A principal vantagem de ter quatro camadas com meta-metamodelo comum é apoiar múltiplos metamodelos e modelos e, assim, garantir extensibilidade, integração e gerenciamento deles.

Todos os metamodelos, padronizados ou personalizados, definidos pelo MOF são posicionados na camada M2. Um deles é a UML e seus *profiles*. Os modelos, representados por conceitos definidos no metamodelo da camada M2, estão na camada M1. Finalmente, a camada M0 contém as instâncias dos modelos do mundo real. O lado esquerdo da Figura 4 apresenta este contexto.

Figura 4 - Modelagem de ontologia no contexto MDA e WS, adaptada de (GAŠEVIĆ; DJURIĆ; DEVEDŽIĆ. 2006)

Outro padrão básico da arquitetura é o XMI – um padrão definido para trocar os metamodelos definidos segundo MOF. Assim, com o compartilhamento do mesmo meta-metamodelo, garante-se a interoperabilidade entre os metamodelos e modelos presentes em diferentes repositórios de ferramentas CASE. XMI permite, por exemplo, representar um diagrama UML de forma neutra e intercambiável entre ferramentas de projeto.

Os Mapeamentos entre metamodelos são construídos com linguagens de transformação. Linguagens de transformação definem operacionalmente a transformação de um modelo fonte em um modelo alvo, ou seja, manipula elementos de modelos, considerando os metamodelos utilizados na construção dos mesmos. Exemplos de ferramentas são ATL (*Atlas Transformation Language*)²⁰, YATL (*Yet Another Transformation Language*)²¹, BOTL (*Basic Object-oriented Transformation Language*) e a XSLT (*XML Stylesheet Language Transformation*). OMG propõe o MOF QVT (*Query View Transformations*) – uma linguagem formal de transformação. As primeiras ferramentas desenvolvidas para apoiar MDA estão concentradas nas ferramentas que apoiam UML. Ferramentas CASE como IBM Rational Rose, Borland Together, Magic Draw, Poseidon for UML, entre outras, têm apoio para modelar CIM e PIM, de acordo com as camadas MOF, e para geração de PSM diretamente com o código da linguagem de programação destino.

MDA e sua arquitetura em camadas providenciam uma base sólida para definição de metamodelos de qualquer linguagem de modelagem. No caso para modelar ontologias definiu:

- ODM (*Ontology Definition Metamodel*);
- *Ontology UML Profile* (OUP) – *profile* para apoiar a notação UML na definição de ontologia;
- Mapeamentos entre OWL e ODM, ODM e *Profile UML* para ontologia e *Profile UML* para Ontologia para outros *profiles UML*.

Estas definições podem ser vistas na Figura 4 e é interessante observar a possibilidade da realização de novos mapeamentos conforme linguagens futuras para ontologia na Web forem surgindo.

A especificação do ODM (OMG, 2007a) traz as definições do metamodelo para ontologia, permitindo representar a sintaxe abstrata das linguagens comumente utilizadas pela comunidade da WS (RDFS, OWL e *Topic Maps*) e os metamodelos da Engenharia de Software (E/R e UML2). O Anexo B detalha os principais elementos destes modelos para contemplar a representação de ontologias.

O Quadro 5 apresenta as principais abordagens envolvendo MDA e ontologias. Nele pode-se observar a presença dos padrões – UML, MOF e XMI – nos metamodelos e

²⁰ <http://www.sciences.univ-nantes.fr/lina/atl/atldemo/>

²¹ linguagem desenvolvida por Octavian Patrascoiu na Universidade de Kent – Reino Unido

no formato compartilhado na grande maioria das abordagens, a predominância de mecanismo programado de transformação e a diversidade de linguagens geradas. Estas abordagens já apontam para a necessidade do uso de padrões de metamodelos.

Abordagem	Metamodelo	Formato Compartilhado	Mecanismo Transformação	Linguagem Gerada
(BACLAWSKI et al., 2002)	UML padrão, Profile UML, e metamodelo baseado no MOF	UML XMI e MOF XMI	-	DAML
(FALKOVYCH; SABOU; STUCKENSCHMIDT, 2003)	UML padrão	UML XMI	XSLT	DAML+OIL
VOM (Visual Ontology Modeler) (CECCARONI; KENDALL, 2003)	Profile UML	IBM Rational Rose	programado	OWL, DAML+OIL, RDFS
GOOD OLD AI ODM (DJURIĆ; GAŠEVIĆ; DEVEČIĆ, 2005)	Metamodelo baseado no MOF e <i>profile</i> UML	ODM XMI UML XMI	XSLT	OWL
AIFB OWL DL (BROCKMANS et al., 2006)	Metamodelo baseado no MOF e UML	UML XMI e MOF XMI	-	OWL DL
CraneField (CRANEFIELD; PAN, 2007)	UML padrão	UML XMI	XSLT	RDFS, classes Java
Protégé ²²	Metamodelo Protégé	Protégé XMI	programado	OWL, RDF(S), DAML+OIL, XML, UML XMI, Protégé XMI, ...
	Padrão UML	UML XMI		
DUET (DAML UML Enhanced Tool) ²³	Profile UML	IBM Rational Rose, ArgoUML	programado	DAML+OIL
Xpetal ²⁴	Padrão UML	IBM Rational Rose e arquivos mdl	programado	RDFS

Quadro 5 - Abordagens MDA e ontologias, adaptado de (GAŠEVIĆ; DJURIĆ; DEVEČIĆ, 2006)

2.5.3 Ferramentas

Existem ferramentas para ontologias em todos estágios do ciclo de vida da ontologia incluindo criação, população, implementação e manutenção, bem como para recuperação do conhecimento, armazenamento e compartilhamento das ontologias (YOUN; MCLEOD, 2006). Estas ferramentas são desde ferramentas de software livre até produtos comerciais, desde simples editores gráficos, passando por validação de consistência até sofisticadas arquiteturas com ambiente de

²² <http://protege.stanford.edu>

²³ <http://codip.grci.com/Tools/Tools.html>

²⁴ <http://www.langdale.com.au/styler/xpetal/>

manipulação completo, apresentando mecanismos de extração de ontologias de diversas fontes, de inferência, de construção de axiomas e integração de ontologias. Dentre elas destacam-se Protégé, OntoEdit²⁵ (atual OntoStudio), WebOnto²⁶, OilEd²⁷, WebODE²⁸ e KAON²⁹.

Protégé, WebODE e OntoStudio são ferramentas populares pois apresentam um modelo de conhecimento flexível e simplificam as operações de importação e exportação das ontologias para diferentes linguagens de representação, respeitando seus formalismos (FREITAS, 2003; DENNY, 2002).

Youn e Mcleod (2006) compararam catorze (14) ferramentas baseadas nos padrões W3C (XML, RDF e RDFS, DAML+OIL e OWL) com objetivos similares nos seguintes aspectos: tipo de representação, apresentação gráfica, verificação de consistência, usuários (número de desenvolvedores), suporte Web e integração, trabalho em colaboração, biblioteca de ontologia, mecanismo de inferência, tratamento de erro, tipo de armazenamento e disponibilidade de uso, cujos resultados foram representados no Quadro 6. Ressalta-se a falta de interoperabilidade entre as ferramentas, percebida pelos diferentes formatos importados e exportados, e de funcionalidades para atender as atividades do ciclo de vida da ontologia, percebida pela inexistência de muitas ferramentas de mecanismos de inferência, tratamento de erros, integração e verificação de consistência. Isto cria alguns problemas na ES como para integrar uma ontologia em bibliotecas de diferentes ferramentas, ou quando duas ontologias construídas em diferentes linguagens de representação precisam fazer uso de ferramentas de junção. O uso de metamodelos padronizados MOF-M2 nas ferramentas resolveria estes problemas, pois possibilita a troca dos modelos (MOF-M1) e das instâncias (MOF-M0) entre as ferramentas.

²⁵ <http://www.ontoknowledge.org/tools/ontoedit.shtml>

²⁶ <http://kmi.open.ac.uk/projects/Webonto>

²⁷ <http://img.cs.man.ac.uk/oil>

²⁸ <http://Webode.dia.fi.upm.es/WebODEWeb/index.html>

²⁹ <http://kaon.semanticWeb.org/>

Exemplos de ferramentas para avaliação, gravação e consulta de ontologias são enumeradas no Quadro 7.

Tipo	Ferramenta	Referência
Avaliação	OntoClean	(WELTY; KALRA; CHU-CARROLL, 2003)
	OntoMetric	(LOZANO-TELLO; GOMEZ-PÉREZ, 2004)
	ODEval	(CORCHO et al., 2004)
	AEON (<i>Automatic Evaluation of Ontologies</i>)	(VÖLKER; VRANDECIC; SURE, 2005)
Gravação e Consulta	Redland	(BECKETT, 2001)
	Inkling	(MILLER, 2001)
	Jena	(MCBRIDE, 2001)
	Sesame	(BROEKSTRA; KAMPMAN; HARMELEN, 2002)
	Cerebra	(BERKOWITZ, 2003)

Quadro 7 - Ferramentas para Avaliação de Ontologias

2.6 Aprendizado de Ontologias

A construção manual de ontologias requer muito esforço dos especialistas do domínio e seus processos consomem tempo e dinheiro. Também é considerada uma tarefa tediosa, tendenciosa ao conhecimento do seu desenvolvedor, não-flexível a mudanças e específica para o propósito que a motivou (GANGEMI et al., 2008; ZAVITSANOS; VOUIROS; PALIOURAS, 2006; SHAMSFARD; BARFOROUSH, 2004; MAEDCHE, 2002).

Por isto existe, dentro da EO, a área denominada Aprendizado de Ontologia (AO) ou *Ontology Learning*: um conjunto de métodos e técnicas usado para construir uma ontologia através de processos automáticos ou semi-automáticos de aquisição de conhecimento via textos, dicionários, bases de conhecimento, dados semi-estruturados e esquemas relacionais já existentes (SPYNS; MEERSMAN; JARRAR, 2002; GÓMEZ-PÉREZ; MANZANO-MACHO, 2003).

Assim, o grande volume de dados disponibilizados pelas organizações, comunidades científicas e indivíduos em diferentes fontes de informação, como na Web ou em banco de dados legados, pode ser transformado em informações úteis ou conhecimento para apoiar os engenheiros de ontologia na aquisição e manutenção de representações de ontologia.

AO tem uma variedade de fontes como esquemas de banco de dados, ontologias já implantadas, textos e recursos em linguagem natural. Quando dados são modelados em esquemas relacionais, o trabalho feito durante a fase de modelagem pode ser

utilizado no AO. Se um esquema de banco de dados existe, então existe uma semântica formal que pode ser combinada em outra mais complexa para integrar um mapeamento semântico (ASTROVA; STANTIC, 2005). Se dados estão baseados em esquemas Web, como nos esquemas XML, ontologias podem ser derivadas destas estruturas. Se dados estão definidos como instâncias, AO pode recuperá-los numa ontologia. Já com os dados semi-estruturados, o objetivo é procurar uma semântica implícita.

AO integra vários métodos de diferentes campos como Aprendizagem de Máquinas, Aquisição de Conhecimento, Processamento de Linguagem Natural (PLN), Recuperação de Informação, Inteligência Artificial, Mecanismos de Raciocínio e Gerenciamento de Banco de Dados.

Para reduzir os esforços na construção de ontologias, muitas abordagens para automação parcial do processo de aquisição do conhecimento têm sido propostas.

2.6.1 Processo Básico

AO é um conjunto de métodos e técnicas para construir qualquer ontologia sem um conhecimento prévio, enriquecendo-a ou adaptando-a a uma ontologia existente de uma forma semi-automática, usando múltiplas fontes de informação. Assim, refere-se à extração de elementos da ontologia de artefatos existentes – modelos já construídos ou textos - e à construção de uma ontologia a partir deles (SHAMSFARD; BARFOROUSH, 2004), e pode ser visto como um *plug-in*³⁰ no processo de desenvolvimento de ontologia. Isto é importante para definir em qual fase pode ser eficiente automatizar a recuperação do conhecimento já representado nas diversas metodologias propostas na Engenharia de Ontologias.

Dado um conjunto de artefatos presentes nos recursos de entrada, um módulo de enriquecimento identifica novos conceitos, enriquece a taxonomia existente incluindo relações entre os conceitos e determina as relações semânticas. A saída desta tarefa é uma ontologia que precisa ser refinada quando ocorrerem inconsistências. São subtarefas da tarefa de enriquecimento de conceitos: a identificação de termos, a construção de taxonomia e a extração de relações semânticas.

³⁰ Pequeno programa auto-executável

A identificação de termos extrai os termos relevantes do corpo do texto como candidatos a conceitos na ontologia. Segundo Lara (2004), a norma ISO 1087 define terminologia como um conjunto de termos, representando o sistema de conceitos de um campo particular. Termos são palavras ou conjunto de caracteres que exprimem um significado único para uma dada comunidade. Um conceito é algo que se tem em mente como um objeto, uma idéia, uma abstração. Um termo lexicaliza um conceito; assim, em muitos casos, um termo indica, ou pode representar, candidatos a conceitos na ontologia. A identificação de termos pode ser subdividida em reconhecimento, classificação e mapeamento de termos. Por exemplo, num conjunto de documentos, identificam-se os seguintes termos: rio, país, cidade, capital e nação.

A construção de taxonomia coloca os candidatos a conceitos numa hierarquia através da inclusão de relacionamentos entre os conceitos.

Seguindo o exemplo acima, obtém-se:

cidade $\subseteq$ país (cidade é um subconjunto de país ou cidade está contida em país)

capital $\subseteq$ cidade (capital é um subconjunto de cidade ou capital está contida em cidade)

A extração de relações semânticas inclui as outras relações não-taxonômicas existentes entre os conceitos. Algumas relações são parte-de (*part-of*), propósito (*X is used for Y*) e ação. Os tipos de relações mais importantes entre conceitos são: a relação de causa (*X leads to Y*) e a relação de atributo (*the X of Y*). Abordagens estatísticas, lingüísticas e aprendizagem de máquinas são propostas.

No exemplo obtém-se:

país $\vee$ nação

fluxo_atraves_de(domínio:rio, limite: país)

E assim os elementos da ontologia são aprendidos progressivamente como observa-se na Figura 5, sugerida por Cimiano (2006) e Zouaq, Nkambou e Frasson (2007). A pirâmide apresentada define, além da hierarquia de elementos aprendidos, a complexidade no aprendizado dos elementos. Assim, quanto mais alto o elemento na pirâmide, maior será a complexidade do algoritmo elaborado para sua obtenção.

Figura 5 – Camadas de elementos da ontologia, adaptada de (CIMIANO, 2006)

Uma ontologia não pode incluir informações contraditórias ou erros de sintaxe senão o mecanismo de raciocínio pode tornar-se infinito. Logo, um processo de avaliação é necessário para julgar se a ontologia obtida será aplicável e representável no domínio de interesse. Este é um processo iterativo e repetitivo pela busca da representação mais precisa dentre as possibilidades do domínio de interesse.

Maedche (2002) no projeto KAON propôs uma visão orientada a processos no ciclo do AO com quatro fases: importar, extrair, aparar e refinar. Ressalta a independência existente entre as fases, ou seja, uma fase não depende necessariamente da outra. A primeira fase consiste em importar e reusar ontologias existentes. Muitas ontologias léxico semânticas existentes - como *WordNet*³¹ e *GermaNet*³² - podem ser utilizadas pelos algoritmos de extração na construção de uma ontologia para um domínio específico. Na segunda fase, algoritmos para extração modelam a maior parte da ontologia desejada a partir de uma base de conhecimento dada. A maior parte das pesquisas em AO dedica-se a esta fase. A terceira fase ajusta, melhora e prepara a ontologia para seu objetivo principal. Nesta fase, estruturas podem ser eliminadas para a ontologia ser mais aderente à aplicação na qual atuará. Na quarta fase é realizado um refinamento da ontologia no sentido de incluir novos domínios na ontologia construída, manter e alterar seu escopo.

³¹ <http://wordnet.princeton.edu/>

³² <http://www.sfs.uni-tuebingen.de/lzd/>

A Figura 6 mostra a arquitetura deste processo com os processos executados pelos engenheiros de ontologia. Os principais componentes são de Gerenciamento Gráfico da Ontologia (1), para Processamento de Recursos (3), a Biblioteca de Algoritmos (6) e a Coordenação (2).

Figura 6 – Arquitetura Conceitual, adaptada de (MAEDCHE, 2002)

O Engenheiro de Ontologia usa o componente Gerenciamento Gráfico (1) para manter a Ontologia de Domínio (5), ou seja, incluir ontologias existentes – entre elas as ontologias léxico semânticas existentes (4) –, visualizar, validar, modificar, controlar versões e avaliar a ontologia obtida.

Os componentes para Processamento de Recursos (3) contêm um conjunto de técnicas para descobrir, importar, analisar e transformar entradas de dados relevantes, presentes nos documentos Web, em esquemas semi-estruturados ou em esquemas de banco de dados e/ou ontologias existentes. Um importante sub-componente é um sistema de PLN. Este sistema gera um conjunto de Dados Pré-processados (7) que serão as entradas para o componente Biblioteca de Algoritmos (6). Este atua como espinha dorsal da arquitetura. Um certo número de algoritmos é providenciado para a extração e manutenção das partes da ontologia contidas na Ontologia de Domínio (5) e para poder combinar os resultados de extração de

diferentes algoritmos de aprendizado, Dados Pré-processados (7). Se muitos algoritmos de extração obtiverem resultados complementares ou sobrepostos, eles são combinados e apresentados para o engenheiro de ontologia, somente uma vez, no Conjunto de Resultados (8).

O componente de Coordenação (2) é utilizado pelo Engenheiro de Ontologia para interagir com os componentes do AO para processar os recursos e utilizar a Biblioteca de Algoritmos. Interfaces de usuário amigáveis são providenciadas pelo Gerenciamento Gráfico (1) e ajudam o Engenheiro de Ontologias a selecionar dados relevantes, aplicar as técnicas para processamento ou iniciar um mecanismo específico de extração. Os resultados são unificados, estruturados e padronizados no Conjunto de Resultados (8) e apresentados ao Engenheiro de Ontologia em diferentes visões da representação da ontologia. Oberle et al. (2005) atualizaram esta arquitetura conceitual, incorporando serviços semânticos. As ferramentas utilizadas na sua implementação são o Text2Onto e o OntoEdit.

A junção de diferentes representações de ontologias obtidas é uma tarefa importante dentro do AO. Vários algoritmos ou processos de extração são aplicados no mesmo conjunto de recursos de entrada ou em conjuntos diferentes, obtendo diferentes representações que precisam ser combinadas e apresentadas numa única representação para o Engenheiro de Ontologias. Um método conhecido para isto é o FCA-Merge e suas ferramentas (CHOI; SONG; HAN, 2006).

AO é uma área de pesquisa emergente e ainda não tem padrões e critérios de avaliação na comunidade de pesquisa para a ontologia gerada e para as subtarefas do processo de aprendizado (ZAVITSANOS; VOUIROS; PALIOURAS, 2006). As ferramentas ainda são imaturas, apesar de necessárias para facilitar o processo de desenvolvimento de ontologias. Novos aspectos metodológicos, técnicas, padrões, ferramentas e critérios de avaliação se fazem ainda necessários para apoiar o AO de forma consistente.

2.6.2 Técnicas

As técnicas aplicadas no AO estão diretamente relacionadas ao tipo dos dados de entrada. Estes dados, ditos relevantes para AO, foram divididos em cinco tipos

principais: ontologias existentes, schemata, instâncias, documentos Web e dados semi-estruturados.

O reuso de ontologias existentes é um ponto crítico na WS pois cada vez mais ontologias estão surgindo e precisam ser adaptadas de um domínio para outro, ou estendidas na sua abrangência.

Os dados na forma de schemata possuem uma semântica formal e são tipicamente representados em modelos de dados como E/R ou diagramas UML. Geralmente, técnicas de Engenharia Reversa com mapeamentos para linguagens de representação de ontologia são utilizadas. Também, inclui-se nesta classificação os documentos XML e XML Schema por apresentarem dados semi-estruturados num esquema rígido e fixo.

Os dados na forma de instâncias consideram as bases de conhecimento existentes junto com a descrição dos conceitos do domínio. Estas bases representam dados relevantes para extrair as regras indutivas descritas para o dado domínio.

Os dados semi-estruturados têm uma estrutura semântica implícita e esta precisa ser descoberta.

Os dados tipo documentos Web estão em textos expressos em linguagem natural. Estes são os mais estudados no AO. Terabytes e terabytes de textos contendo opiniões, idéias, fatos e informações de toda sorte esperam serem encontrados em algum tipo de busca ou serem anotados em documentos similares para facilitar sua recuperação (CIMIANO, 2006).

Os dois últimos tipos de dados – semi-estruturados e documentos Web - têm como principais técnicas aplicadas a estatística, a aprendizagem de máquinas e as técnicas lingüísticas.

As técnicas estatísticas são baseadas na freqüência de termos no documento ou no conjunto de documentos. O interesse pode estar nos termos com alta ou baixa freqüência. Essas técnicas são utilizadas, também, para detectar sinônimos, identificar termos em grupos e descobrir novos termos. Às vezes, são utilizadas junto com um dicionário para aquisição do conceito, como, por exemplo, em pesquisas combinadas com o WordNet.

A aprendizagem de máquinas é usualmente projetada para uma classe de entidades específica e integra o reconhecimento e a classificação de termos. As funcionalidades padrão para este propósito são ortografia, prefixo e sufixo das palavras.

As técnicas lingüísticas requerem o PLN. Dado um conjunto de documentos, a primeira abordagem é dividi-lo em segmentos maiores (parágrafos, títulos) usando uma lista de frases, unidades etc., para detectar a presença de termos candidatos. Estas técnicas podem ser aplicadas individual ou conjuntamente no AO. Maiores detalhes das principais técnicas podem ser encontradas no Anexo C.

Já para os schematas, existe a engenharia reversa de bases de dados relacionais: uma abordagem baseada na idéia de que a semântica de uma base de dados relacional pode ser inferida, sem uma análise explícita do esquema relacional, através das tuplas e consultas dos usuários (ASTROVA, 2004). O modelo de dados e a representação da ontologia têm sobreposição na expressividade (classes ou conceitos em ontologias correspondem a entidades no modelo E/R; atributos e relacionamentos no modelo E/R correspondem às propriedades nas linguagens de ontologias, entre outras) e algumas diferenças, geralmente atribuídas no uso de um esquema de banco de dados e de uma ontologia. Ontologias têm objetivos como interoperabilidade, busca, e especificação de software. O objetivo inicial de um esquema de banco de dados é estruturar um conjunto de instâncias para consultar um único banco de dados. Estas diferenças impactam fortemente na aquisição de restrições. Para ontologias, restrições são chamadas axiomas. Seu objetivo principal é expressar um significado lido, automaticamente, para apoiar o mecanismo de raciocínio automatizado acurado. Este raciocínio pode também garantir a integridade de instâncias na base de conhecimento. Para banco de dados, o objetivo principal de restrições é garantir a integridade de dados (isto é, as instâncias). Todas as restrições de integridade podem também ser utilizadas para otimizar consultas e ajudar inferências humanas no significado de termos.

O número de algoritmos associados a cada técnica é grande e aumenta quando ocorre a combinação deles. Isto ocorre por ser esta uma área constantemente pesquisada devido, principalmente, às características lingüísticas de cada língua e aos resultados desejados em cada contexto.

2.6.3 Ferramentas

Muitas abordagens para AO foram propostas nos últimos anos, diferindo umas das outras em alguma dimensão. Shamsfard e Barforoush (2004) classificaram seis

dimensões principais e algumas subcategorias numa análise realizada com sete sistemas de aprendizado, considerados representativos na época da pesquisa. As dimensões de um *framework* para AO foram:

- ☐ Elementos aprendidos (conceitos, relações, axiomas, regras, instâncias, categorias sintáticas, entre outros);
- ☐ Ponto de início (conhecimento prévio e seu tipo, e linguagem de entrada);
- ☐ Pré-processo (processamento lingüístico);
- ☐ Método de aprendizado:
 - ☐ Categoria de aprendizado (supervisionado ou não, *on-line* ou não);
 - ☐ Abordagem de aprendizado (estatística ou simbólica, lógica, lingüística, métodos híbridos, entre outras);
 - ☐ Tarefas para aprendizado (classificação, agrupamento, aprendizado por regras, formação de conceitos, população da ontologia);
 - ☐ Grau de automação (manual, semi-automática, cooperativa, totalmente automática).
- ☐ Resultado (ontologia ou estrutura intermediária);
- ☐ Métodos de avaliação (do método e da ontologia resultante).

O Quadro 8 apresenta o resultado obtido.

Choi, Song e Han (2006) estudaram ferramentas para junção (*merge*) de ontologias, ou seja, ferramentas de AO para combinar ontologias distribuídas e heterogêneas, e destacaram características como dados de entrada, algoritmos utilizados e interação com usuário, bem como resultados alcançados. Neste estudo destacaram MOMIS (*Mediator Environment for Multiple Information Sources*), LSD (*Learning Source Descriptions*), Ctxmatch, Glue, MAFRA (*MApping FRAmework*), LOM (*Lexicon-based Ontology Mapping*), ONION (*ONtology compositiON*), Prompt e FCA-Merge. Tanto em Shamsfard e Barforoush (2004) como em Choi, Song e Han (2006), cada sistema de aprendizado apresenta uma peculiaridade própria seja nos elementos aprendidos, no tipo de recurso de entrada, no método de aprendizado ou linguagem destino. Por exemplo, o sistema de aprendizado Hasti aprende todos os elementos de uma ontologia, tem como entrada (ponto de início) textos escritos em persa e expressa o resultado na linguagem *Knowledge Interchange Format* (KIF). As diferentes linguagens de entrada mostram sistemas de aprendizado limitados aos ambientes de sua construção e os resultados expressos em diferentes linguagens mostram a falta de adoção de padrões na representação final. Talvez isto tudo resulte do interesse maior dos pesquisadores no sucesso da técnica de aprendizado empregada nos tipos de entrada selecionados, em detrimento do compartilhamento dos resultados da ferramenta criada com outras.

Gómez-Pérez e Manzano-Macho (2003) também detalharam e avaliaram os métodos e as ferramentas para AO e enfatizaram a classificação das ferramentas por tipo de dados de entrada:

- textos: Asium, Caméléon, Corporum-Ontobuilder, DOE (*Differential Ontology Editor*), KEA (*Keyphrases Extraction Algorithm*), LTG (*Language Technology Group*), Mo'K, OntoLearn, Prométhée, SOAT (*Semi-Automatic Domain Ontology Acquisition Tool*), SubWordNet *Engineering*, Svetlan, Terminae, Text-To-Onto (atual Text2Onto), Welkin, WOLFIE (*WOrd Learning From Interpreted Examples*);
- dicionários: SEISD (*System for Extraction of Semantic Information from Dictionaries*), DODDLE (*Domain Ontology Rapid Development Environment*);
- bases de conhecimento: nenhuma ferramenta relevante;
- dados semi-estruturados: OntoBuilder;
- schematas: nenhuma ferramenta relevante citada, porém existe a Eronto desenvolvida por Upadhyaya, Kumar e Eronto (2005).

2.7 Discussão

A WS é uma tecnologia considerada nova mas já apresenta uma diversidade de linguagens, ferramentas, técnicas e metodologias, tanto para sua representação, quanto para sua utilização. O W3C definiu rapidamente uma arquitetura técnica padrão para suas camadas, protocolos e linguagens a fim de organizar e facilitar a interoperabilidade das ontologias. Porém, como a semântica é uma área pesquisada há mais tempo, traz consigo muitas formas de representação que são aproveitadas na representação das ontologias, principalmente, as associadas à representação do conhecimento. Assim, muitas destas representações foram implementadas em editores de ontologias, não necessariamente nos padrões RDFS e OWL propostos pelo W3C. Muitas pesquisas na WS tentam aproveitar o resultado destas pesquisas, convertendo-o para os padrões definidos no W3C.

As metodologias, embora com algumas atividades consideradas clássicas - como análise de domínio, conceitualização e implementação – sofrem com o dilema da construção centralizada ou colaborativa. Uma ontologia para ser compartilhada precisa ter o consenso dos usuários no domínio de sua utilização. Assim, a construção colaborativa procura fazer uma discussão sobre as conceitualizações entre pessoas em diferentes localizações e isto demanda um tempo maior para sua construção.

A ES também aborda o tema ontologia principalmente sob a perspectiva do OMG. O OMG tornou-se um importante grupo de divulgação de tecnologias para interoperabilidade em sistemas distribuídos, com a divulgação da UML como padrão para representação de artefatos técnicos dos sistemas. A UML é um padrão aceito e utilizado por toda a comunidade técnica. No rastro da UML existe a MDA e seus padrões MOF e XMI. O MOF, ao criar suas camadas para meta-metamodelos, metamodelos e modelos, permite que a abstração representada em ferramentas possa ser aproveitada em outras ferramentas, utilizando mapeamentos entre metamodelos para arquivos XMI. Assim, estes padrões suportam outros processos como avaliação da qualidade, facilidade de reuso e manutenção dos sistemas desenvolvidos. O ODM é a iniciativa dentro do OMG para incorporar a representação conceitual da ontologia na MDA.

A principal vantagem em utilizar UML na representação conceitual da ontologia é habilitar rapidamente profissionais no conhecimento e técnicas da EO. O *profile* UML

traz os novos elementos e notações necessários para apoiar a representação da ontologia, os mapeamentos para os principais metamodelos existentes na EO (OWL e Topic Maps) e as soluções para resolver a maioria das divergências de representatividade entre os elementos de uma notação e outra.

A WS precisa construir ontologias rapidamente e para agilizar este processo surgiu o AO com muitas técnicas e ferramentas, de acordo com o elemento a ser aprendido – conceito, relação, função e axioma. O PLN e Aprendizagem de Máquinas são as técnicas mais pesquisadas e apresentam um grande número de algoritmos e combinações deles para realizarem sua tarefa. A eficiência do algoritmo e sua implementação é o destaque das ferramentas construídas. Poucas pesquisas preocupam-se em representar os elementos aprendidos nos padrões W3C (RDF, OWL).

A EO encontra dificuldades durante o desenvolvimento de diferentes tipos de ontologias, entre elas:

- ☐ custo e tempo de desenvolvimento principalmente quando ocorrer inferências automáticas e formais;
- ☐ consenso entre usuário, especialista do domínio e engenheiro de ontologias pela falta de suporte para compartilhar e reusar conhecimento acumulado sem falar que muitos envolvidos não compartilham das mesmas suposições;
- ☐ falta de ferramentas robustas para transportar a representação entre diferentes linguagens, incluindo métodos para converter estruturas conceituais semi-estruturadas para OWL e RDFS.

Os maiores problemas na construção e uso de ontologias são os obstáculos na aquisição de conhecimento e o tempo consumido na construção e na integração de ontologias para vários domínios ou aplicações. A pesquisa realizada por Simperl e Tempich (2006) apontou isto somado a falta de uma abordagem sistemática para construção de ontologia, ou seja, nenhuma das metodologias existentes está sendo utilizada em grande escala. Também constataram que os grupos de pessoas envolvidas na construção das ontologias são pequenos e existe dificuldade na compreensão das linguagens de representação.

O universo da WS e ontologias pode ganhar força e projeção com a utilização e presença de padrões consolidados e conhecidos da ES elaborados pelo OMG aliados aos do W3C.

3 EDUCAÇÃO E WEB SEMÂNTICA

Educação e TIC (Tecnologia, Informação e Comunicação) andam juntas a algum tempo, desde aplicações para gestão do ambiente ensino-aprendizagem até aplicações para gerenciamento de aprendizagem baseados na Web, incluindo aqui todos tipos de aplicações para Educação a Distância (EAD) e classes virtuais com o uso de tutoriais inteligentes e aprendizagem adaptativa.

A rápida evolução da tecnologia traz consigo novos desafios para o domínio da Educação e, neste momento, é importante pensar no re-planejamento das relações entre educandos e educadores, sociedade e setor educativo, tecnologia e educação (CAMPOS; DA ROCHA; DE CAMPOS, 1998; BELLONI, 2003).

Neste contexto apresenta-se o cenário no domínio da Educação com destaque para os estudos com objetos de aprendizagem e projeto instrucional, a Web Semântica na Educação (WSE) com o foco em metadados e ontologias e os principais projetos educacionais em curso.

3.1 Cenário Atual

Especialistas em Educação, juntamente com especialistas em TIC, têm desenvolvido ao longo do tempo³³ aplicações para uso no ambiente educacional, tanto para atender as necessidades educacionais, como para apoiar a evolução tecnológica. Hew (2004) cita exemplos como a máquina para corrigir testes de múltipla escolha com cartões perfurados do Dr. Sidney Pressey (1924), a máquina de ensinar com instruções programadas idealizada por Skinner (1950) e as aplicações baseadas no conceito de instrução auxiliada por computador como o programa Platô (1960). As novas modalidades de ensino a distância, simulação, jogos educacionais e outros surgiram em 1980. Já com a Web, pode-se construir ambientes de aprendizagem cooperativos e interativos, ambientes virtuais de aprendizagem, comunidades virtuais e simuladores, entre outros. Todos utilizando recursos educacionais e incentivando trocas de experiências e compartilhamento de informações, com o objetivo de favorecer o ensino e a aprendizagem através do uso intenso de TIC.

³³ <http://www.aect.org/standards/history.html>

Koper (2004) resumiu e reagrupou os problemas e necessidades do ensino-aprendizagem em três dimensões de mudança: demanda social, processos de ensino-aprendizagem e organização das instituições educacionais, conforme apresentado no Quadro 9. Isto foi resultado de pesquisas realizadas por Howell, Williams e Lindsay (2003) e Merrill (2002). Os primeiros apresentaram o resultado da revisão de literatura realizada com publicações mais referenciadas nos últimos três anos da pesquisa e relevantes para EAD na comunidade do ensino superior. O último realizou uma pesquisa para justificar a hipótese que falhas na implementação de programas e práticas instrucionais causam um decréscimo na aprendizagem.

Dimensão	Problema / Necessidade
I. Mudanças na demanda social	1. A atual infra-estrutura educacional das instituições de ensino de alto nível não consegue acompanhar o crescimento populacional em idade escolar e a necessidade crescente de aprendizagem ao longo da vida
	2. Conhecimento e informação crescem exponencialmente e a aprendizagem ao longo da vida transforma-se num processo competitivo
	3. Educação é cada vez mais contínua, sem intervalos de tempo entre o colégio, a universidade e os estudos complementares
II. Mudanças no processo ensino e aprendizagem	1. Instrução é centrada no estudante, não linear e auto dirigida
	2. Existe uma necessidade crescente de novas estratégias de aprendizagem e ensino, baseadas nas novas pesquisas de projeto instrucional e na exploração das potencialidades da tecnologia
	3. Aprendizagem é mais eficaz quando os estudantes estão engajados em resolver problemas do mundo real; ambientes de aprendizagem precisam ser projetados para apoiar esta abordagem
	4. Estudantes demandam por maior flexibilidade e compram cursos para atender seus cronogramas e circunstâncias
	5. Os perfis dos estudantes universitários estão mudando
	6. A ênfase acadêmica desloca-se da conclusão do curso para a competência provida por ele
	7. A necessidade para o desenvolvimento, a sustentação e o treinamento da faculdade está crescendo
	8. Os instrutores de cursos a distância podem sentir-se isolados
III. Mudanças na Organização das Instituições Educacionais	1. Existe uma tendência à descentralização na estrutura organizacional
	2. As parcerias e a terceirização para educação superior estão aumentando
	3. As taxas de retenção e o cumprimento de prazos para conclusão dos cursos afetam administradores, membros da instituição de ensino e estudantes
	4. A distinção entre a distância e o local da aprendizagem está desaparecendo
	5. Colaboradores da faculdade almejam a redução da carga de trabalho e aumento da compensação para cursos a distância
	6. Os papéis tradicionais da instituição de ensino estão desaparecendo ou mudando

Quadro 9 - Dimensões e problemas do ensino e aprendizagem, adaptado de (KOPER, 2004)

No Brasil, quanto às mudanças na demanda social, não será necessário apenas oferecer à força de trabalho oportunidades de formação contínua de atualização

exigida pelas mudanças econômicas e tecnológicas, como também será imprescindível elevar o nível de educação básica dos trabalhadores. Existem, também, segundo Belloni (2003), os fatores do acesso à tecnologia ser desigualmente distribuído em termos sociais e das habilidades requeridas serem pouco desenvolvidas na população em geral, seja em razão dos baixos níveis de escolaridade, seja pela falta de qualidade do ensino. Já as mudanças no processo ensino-aprendizagem requerem políticas públicas decisivas de investimento em educação superior e formação continuada do pessoal docente, não apenas para a integração de tecnologias em seu ensino, mas para uma verdadeira transformação de seu papel e funções. As mudanças na organização das instituições educacionais com os efeitos da globalização no campo da EAD tendem a ser mais negativos do que positivos, pois, corre-se o risco de importação e/ou adaptação de tecnologias caras e pouco apropriadas às necessidades e demandas da nossa população.

Em termos gerais, a mudança educacional busca aumentar a eficácia, a flexibilidade e a acessibilidade e a atratividade da educação, e diminuir a carga de trabalho da equipe de colaboradores, obtendo como resultado final a diminuição dos custos institucionais.

A tecnologia educacional baseada no paradigma tradicional tem um relacionamento linear e hierárquico, envolvendo a pré-seleção do conteúdo, enquanto que na baseada nos paradigmas atuais com enfoque construtivista os alunos selecionam e desenvolvem suas próprias estratégias e são encorajados a buscar novos domínios do conhecimento. Um aspecto fundamental para a reestruturação das instituições de ensino é o foco na aprendizagem de tarefas relevantes, isto é, tarefas que tenham importância e utilidade no mundo real, integradas ao currículo escolar, com diferentes níveis de complexidade e que permitam aos estudantes a seleção de atividades com diferente grau de dificuldade de acordo com o seu interesse e capacidade (CLARK; PARSIA; HENDLER, 2004).

A era do conhecimento precisa formar trabalhadores com habilidades para pensar rápido, ser altamente criativo, agir em colaboração, e ter uma interpretação crítica. Além disso, o sucesso econômico tornou-se cada vez mais dependente do ajuste flexível e antecipado de modelos, de processos e de comportamentos do negócio. De acordo com estudos, 50% de todas as habilidades do trabalhador do conhecimento se tornam obsoletas no período de três a cinco anos (ARROYO et al., 2006). Somado a isso existem diferentes tecnologias inseridas no contexto

educacional de acordo com o tipo de aprendizagem proposto. Campos, Da Rocha e De Campos (1998), reportando-se a vários autores, relacionaram os tipos de aprendizagem e o uso de novas tecnologias, conforme mostra o Quadro 10.

Tipo de Aprendizagem	Descrição	Uso de novas tecnologias
Descoberta imprevista	Aprendizagem não planejada, nenhuma instrução é envolvida diretamente	Busca livre na Internet
Descoberta por livre exploração	Os macros objetivos são fixados e os alunos ficam livres para explorar métodos, sub-objetivos etc.	Busca em programas hipermídia ou Web, com tema definido
Descoberta guiada	Os objetivos de cada passo da aprendizagem são fixados. O aprendiz é livre para explorar métodos, mas com guia e ajuda em cada estágio.	Hipermídia adaptativa, privilegiando o interesse dos alunos
Descoberta adaptativa	Abordagem cibernética, guia e reforço corretivo com estrutura de diagnóstico	Hipermídia adaptativa, baseada no estereótipo dos alunos e definição de nós a serem visitados
Descoberta linear / intrínseca	Direcionada rigidamente; guia e reforço são pré-programados, baseados no estudante típico	Sistemas hipermídia, com excursão definida
Exposição indutiva	O aluno recebe o argumento e não tem que descobrir a regra. Pode ser programado.	Multimídia
Exposição dedutiva	A compreensão do problema é mostrada pela habilidade de aplicá-lo a exemplos, pode ser programado.	Multimídia adaptativa
Aprendizagem de recepção direcionada (exercício e prática)	Aprendizagem de fatos, sentenças e operações, sem entender os conceitos envolvidos. Pode ser programado. Memorização.	Apresentação multimídia
Aprendizagem de recepção imprevista	Fatos e observações, originalmente não planejados, fornecidas por professores, outras fontes e estudantes.	Ferramentas de trabalho cooperativo

Quadro 10 - Tipos de aprendizagem e uso de novas tecnologias
(CAMPOS; DA ROCHA; DE CAMPOS, 1998)

Para apoiar essas propostas foram desenvolvidas inúmeras ferramentas já apresentadas na seção 1.2. Porém, esse cenário traz como consequência, a necessidade de novos métodos pedagógicos para suprir as exigências na educação e treinamento diários. Mais especificamente, requer “sistemas baseados em semântica e no contexto para adquirir, organizar, personalizar, compartilhar e usar o conhecimento dentro da Web, e para conseguir a interoperabilidade semântica entre recursos da informação e serviços heterogêneos” (ARROYO et al., 2006).

A Web Semântica na Educação (WSE) provê significado para representar, organizar e interconectar conhecimento dos educadores, estudantes e autores de material educacional numa forma processável e compreensível por máquinas (DEVEDŽIĆ, 2006). A interoperabilidade semântica permite compartilhar as diferenças entre sistemas de informação em dois níveis: de acesso e de significado. No nível de acesso os limites do sistema e da organização são cruzados pela criação de

interfaces padronizadas para compartilhar os serviços internos de sistemas fracamente acoplados. No nível de significado, acordos sobre os dados transportados são feitos para permitir sua interpretação correta.

Koper (2004) reduz suas expectativas quanto à WSE em dois grupos de pessoas, relacionados com o processamento automático da informação:

- Equipe de funcionários: pode ser auxiliada na realização de suas tarefas, que podem ser realizadas com maior flexibilidade, pelo uso de configurações *on-line* eficientes e menos isoladas; entre elas a abertura de cursos *on-line*, tarefas de suporte ao estudante, tarefas de avaliação e gerenciamento de cursos e tarefas administrativas, como configurar novas instâncias de cursos.
- Pessoas em diferentes papéis como estudantes, tutores e responsáveis pelo conteúdo: podem ser apoiadas na realização de suas tarefas por meio de espaços de aprendizagem mais amplos, distribuídos, compartilhados por inúmeros atores, com multirecursos de aprendizagem e configurados para oferecer oportunidades de aprendizagem ao longo da vida.

Já Stojanovic, Staab e Studer (2001) apresentam um sumário das possibilidades da WSE na realização das expectativas da educação a distância, que se encontram resumidas no Quadro 11.

Expectativa	Web Semântica
Entrega	Itens de conhecimento (material de aprendizagem) são distribuídos na Web, mas eles estão ligados a ontologias, geralmente, acordadas
Acordos	Agentes de software precisam ter uma linguagem de serviço acordada para garantir a coordenação entre os agentes e a entrega pró-ativa dos objetos de aprendizagem, de acordo com o contexto atual do problema
Acesso	Usuários podem descrever suas características (objetivo de aprendizagem, conhecimento prévio etc.) e realizarem consultas semânticas ao material de aprendizagem, de acordo com seu perfil
Simetria	A WS pode transformar-se em uma plataforma de integração de todos os processos de negócio de uma organização, incluindo as atividades de aprendizagem
Modularidade	A entrega ativa da informação (baseada em agentes personalizados) cria um ambiente dinâmico de aprendizagem, totalmente integrado ao processo de negócios
Autoridade	A WS será tanto quanto possível descentralizada, o que garante um gerenciamento do conteúdo cooperativo e efetivo
Personalização	Um usuário (utilizando seu agente personalizado) procura por material de aprendizagem personalizado para suas necessidades e uma ontologia faz a ligação entre as necessidades do usuário e as características do material de aprendizagem
Adaptação	A WS garante o uso de conhecimento distribuído de diversas formas, através da anotação semântica do conteúdo

Quadro 11 - Web Semântica na Educação e suas expectativas ,
baseado em (STOJANOVIC; STAAB; STUDER, 2001)

O domínio da Educação tem excelentes oportunidades para fazer uso da arquitetura oferecida pela WS, pois a Web já é uma fonte de referência para aprender sobre pessoas, lugares e assuntos diversos. A Web, através de suas páginas interligadas, pode levar o indivíduo de uma simples curiosidade sobre triângulo às pirâmides do Egito ou às fantásticas histórias do Triângulo das Bermudas. Já para educar um indivíduo, não basta ele aprender algo, é preciso ele mudar o comportamento a partir do aprendido e para isso é necessário aumentar a interatividade na Web, no sentido de interferir e transformar o aprendido. E isto exige investimentos em infra-estrutura como redes de comunicação, servidores e estações de trabalho, em preparação dos profissionais envolvidos tanto nas novas tecnologias como nas novas abordagens pedagógicas e em construção e disponibilização de recursos educacionais, entre outros. A base teórica está fundamentada porém seu uso efetivo pode demorar algum tempo.

Para dar à WSE uma perspectiva pedagógica, considerando os efeitos que a Web pode ter na construção e no desenvolvimento do conhecimento do indivíduo, duas áreas importantes de pesquisa na Educação são: objetos de aprendizagem e projeto instrucional, que serão discutidas a seguir.

3.1.1 Objetos de Aprendizagem

Rossano et al. (2005) definem Objetos de Aprendizagem (OA) como sendo uma forma contemporânea de organizar conhecimento e informação, como outras formas historicamente conhecidas: a mitologia, a narrativa de poemas e os livros. Logo, OA podem ser livros, documentos Web, lições num curso tradicional, videoconferências, jogos, entre outros.

Para o *Learning Technology Standards Committee* (LTSC) do IEEE, um objeto de aprendizagem é: "qualquer entidade, digital ou não digital, que possa ser usada para aprendizado, educação e treinamento."

Segundo Araújo (2003) e Wiley (2001) não existe uma definição de consenso entre os autores e instituições que pesquisam sobre OA e é melhor focar nas características deles como número de elementos combinados, tipos de objetos contidos, reusabilidade, facilidade de pesquisa e atualização, dependência entre

objetos, tipo de lógica contida no objeto, os contextos de aplicação, modularidade e interoperabilidade, entre outras.

OA envolvem habilidades de muitos profissionais, incluindo professores, cientistas da computação, pedagogos, projetistas instrucionais e desenvolvedores (ROSSANO et al., 2005). Isto leva a três visões: pedagógica - educadores, pragmática - professores e projetista instrucional e técnica - cientistas da computação, desenvolvedores e engenheiros de ontologia.

A abordagem pedagógica permite aos pesquisadores construir ferramentas, modelos ou linguagens, para garantir os aspectos pedagógicos da aprendizagem, enfatizando a abstração aplicada nas diferentes situações e a solução do problema para uma situação específica.

A abordagem pragmática permite aos pesquisadores buscar soluções para problemas relativos ao uso de OA com LMS, determinando regras e princípios para sua integração e garantindo o diálogo entre a abstração e a aplicação.

A abordagem técnica permite aos pesquisadores solucionarem os problemas de construção de OA interoperáveis e apoia, tecnologicamente, os ambientes de aprendizagem. Para que um objeto de aprendizagem possa ser recuperado e reutilizado, é preciso que esse objeto tenha preenchido seus metadados e seja armazenado em um repositório compartilhado ou *Learning Object Repository* (LOR). Assim existem LOR contendo somente metadados como Merlot e outros contendo metadados e OA como *ARIADNE Knowledge Pool System* (DOWNES, 2004).

A consciência do contexto é crucial no projeto dos OA, uma vez que a informação requerida para realizar uma tarefa pode ser dinamicamente inferida de seu ambiente. Neste caso, os OA incluem, não somente o conteúdo didático do curso, mas também informações sobre o contexto no qual eles serão utilizados.

Araujo (2003) cita o desenvolvimento de importantes pesquisas, visando busca, recuperação e adaptação de materiais didáticos para desenvolver aplicações educacionais baseadas em ontologias. Esta idéia possibilita que os materiais de aprendizagem tornem-se estruturados e organizados podendo assim ser disponibilizados na Web em diferentes formatos (hipertexto, vídeo, animações etc.), utilizando as tecnologias que formam a base da WS, porque OA são entidades acessíveis, reutilizáveis e interoperáveis.

A reusabilidade dos OA dependerá da adaptação do conteúdo de aprendizagem para diferentes cenários de aprendizagem, da facilidade para acessar o

conhecimento relevante no contexto desejado, da garantia de metadados menos ambíguos e da expansão dos metadados padronizados para descrever material de cursos como bibliografia, perguntas e respostas freqüentes, e regras para avaliação. Gasévić et al. (2005) distinguem três tipos de ontologias para promover a reusabilidade dos OA: de metadados (OM), de estrutura do conteúdo (OEC) e do domínio ou conteúdo (OD). A Figura 7 apresenta a visão esquemática dessas ontologias.

Figura 7 – Ontologias para Objetos de Aprendizagem, adaptada de (GAŠEVIĆ et al., 2005)

A OM refere-se a uma representação do Metadado do Objeto de Aprendizagem e corresponde à camada inferior do Objeto de Aprendizagem na Figura 7. A OEC serve para explicitar o Conteúdo do Objeto de Aprendizagem enquanto a OD serve para formalizar uma conceitualização compartilhada entre domínios (OD₁, OD₂, OD₃ até OD_n). A OEC permite reutilizar partes dos OA, os seus componentes. Esta definição da estrutura permite a quebra dos OA em unidades de conteúdo.

O objetivo final dos OA é promover a aprendizagem (KOOHANG; DU PLESSIS; BRITZ, 2006). Mas, sem uma estratégia instrucional adequada, os OA não atingem este objetivo. OA precisam apresentar seu conteúdo para os estudantes de uma maneira pedagógica e efetiva. Os objetos sozinhos não promovem a aprendizagem. Eles precisam facilitar, antes de qualquer coisa, a ação de aprendizagem.

Os OA têm recebido muita atenção dos pesquisadores no domínio da Educação e da Tecnologia desde a definição de padrões até a arquitetura de ambientes para melhor implementá-los - de uma definição simples de conteúdo a estruturas complexas envolvendo várias ontologias para seu reuso. Mesmo assim, a construção ou utilização, efetiva e massiva, dos OA ainda não é uma realidade na grande maioria das instituições educacionais ou dos sistemas educacionais construídos recentemente pois requer, além dos investimentos em tempo, recursos

humanos e tecnológicos, fundamentos pedagógicos consistentes para preencher as infinitas situações, espaços e contextos da educação presencial.

3.1.2 Projeto de Aprendizagem

Existe uma grande diferença entre a representação do conhecimento em OA e a construção do conhecimento (CARTELLI, 2006). A construção do conhecimento considera três aspectos:

- ☐ a interação pessoal com a realidade e a construção de esquemas mentais para interpretação de fenômenos, que dela resulta;
- ☐ a interação social entre os assuntos em uma comunidade conhecida;
- ☐ a construção de modelos sociais e do conhecimento, bem estruturados e disciplinados.

Além disso, outras questões são apresentadas por diferentes autores dentre as quais se destacam a importância da natureza (hereditariedade) e da experiência; ainda, algumas coisas são aprendidas melhor com um roteiro programado e outras num contexto significativo e dinâmico; podem existir diferentes objetivos de ensino para um mesmo tópico e um tópico pode exigir diferentes métodos instrucionais.

As teorias da aprendizagem descrevem a forma pela qual uma pessoa aprende e as correntes mais freqüentemente mencionadas são behaviorismo - respostas específicas a estímulos também específicos -, cognitivismo - ligação da nova informação com a informação já existente na memória - e o construtivismo - interpretações do mundo baseadas na experiência anterior e suas inter-relações com outras pessoas.

O projeto educacional define os objetivos educacionais, as atividades constantes no curso e a forma de avaliação, apoiado em uma teoria de aprendizagem. A seleção da melhor solução para o projeto educacional é um problema que envolve princípios sócio-culturais do projetista, fatores externos impostos pelo ambiente e as habilidades consideradas para o aprendiz.

Já o modelo pedagógico contém o conhecimento necessário para tomar decisões sobre quais atividades de ensino se deve empregar dentre aquelas disponíveis no projeto educacional. Define a forma como uma classe de alunos pode alcançar objetivos educacionais em um dado contexto e domínio de conhecimento. Um

modelo descreve, por exemplo, como um adulto aprende espanhol como segunda língua, como engenheiros aprendem habilidades matemáticas.

Por sua vez o projeto instrucional fornece diretrizes concretas sobre como facilitar a ocorrência de certo modelo pedagógico (prescritivo), enquanto teorias de aprendizagem e projetos educacionais fornecem os princípios subjacentes sobre o porquê estas prescrições são úteis (descritivo). Assim, ele liga a teoria de aprendizagem e a prática educacional (HOWELL; WILLIAMS; LINDSAY, 2003).

O projeto instrucional também é conhecido como Projeto de Aprendizagem (PA) e é uma aplicação concreta de um modelo pedagógico para um grupo específico, com objetivos específicos e num domínio específico. O PA especifica o fluxo de trabalho e conteúdo necessário para o processo de ensino e aprendizagem: quais papéis realizam quais atividades, usam-se quais recursos e serviços, em qual ordem eles são aplicados para atingir os objetivos de aprendizagem, considerando a melhor maneira e as diferenças individuais.

PA inclui todas as formas de aprendizagem flexíveis e altamente complexas tanto para encontros de aprendizagem a distância como presenciais (BERGGREN et al., 2005).

Uma unidade de aprendizagem ou UoL (*Unit of Learning*) do PA inclui uma coleção de atividades que pode ter uma seqüência pré-determinada fixa, uma seqüência condicionada ou nenhuma seqüência. Os professores podem compartilhar OA ou UoL, inicialmente entre colegas, depois com a equipe disciplinar, entre departamentos e entre instituições. PA torna-se assim uma notação, um padrão proposto para modelar cenários de aprendizagem. No Anexo C discutem-se propostas neste sentido. Em particular, maior destaque está sendo dado ao IMS LD em vista do envolvimento do Interlab com essa instituição e com o projeto TIDIA-Ae da FAPESP.

Um processo fixo de PA é útil em algumas situações, como no behaviorismo e cognitivismo, mas em outras situações, como no construtivismo, pode dificultar a adaptação de UoLs a circunstâncias imprevistas, ocorridas durante a execução do projeto, particularmente quando ocorrem após o início das atividades. Também, muitos instrutores que usam um LMS para acompanhar suas aulas presenciais constroem seus cursos semana a semana, re-projetando o PA durante ou após a aula. Por isso muitos preferem capturar o PA após ele ter sido exercitado e então compartilhá-lo com a comunidade de professores.

De fato, o projeto educacional é necessário para guiar o processo de autoria de materiais de aprendizagem que efetivamente ajudem o estudante a atingir os objetivos almejados (ULRICH et al., 2005).

O uso de princípios de instrução independentes dos OA garante compor o material do professor com base na estratégia pedagógica e sempre tendo em mente por que uma decisão educacional foi tomada. Uma estrutura de composição clara permite que o sistema acesse os recursos de aprendizagem do PA e reuse todos os recursos disponíveis. Um projetista precisa procurar por material de treinamento aderente aos princípios pedagógicos adotados. Assim, o PA precisa fornecer suporte ao processo de autoria e para composição automática de OA.

A integração das teorias de aprendizagem, PA, OA e tecnologia é como a construção de uma casa. Esta casa precisa ter fundações sólidas (teorias de aprendizagem e projeto educacional), um projeto arquitetônico (modelo pedagógico e projeto de aprendizagem), componentes pré-fabricados de concreto, elétricos e hidráulicos (OA) e alta qualidade de acabamento (tecnologia).

3.2 Metadados e ontologias educacionais

Diante da variedade de sistemas educacionais presente nas instituições de ensino ou no mundo Web, torna-se necessário garantir a interoperabilidade dos recursos educacionais neste universo. Padrões para recursos de aprendizagem são abundantes e cobrem todos os aspectos dos sistemas educacionais, desde a representação, até o empacotamento e a publicação dos OA, passando pelos metadados para descrever os OA, os PA, as entregas da educação, os procedimentos de avaliação e os perfis de estudantes. Além disso, há preocupação com as arquiteturas das aplicações para gerenciamento e aprendizagem educacional.

Padrões são importantes porque estabelecem os princípios para a aprendizagem, organizando os recursos de aprendizagem e o desenvolvimento de aplicações educacionais. Regulam também a interoperabilidade entre as aplicações e permitem o intercâmbio e o reuso de recursos de aprendizagem através de diferentes sistemas, apesar da heterogeneidade dos formatos e das descrições dos metadados entre os domínios.

A experiência na construção de aplicações de educação a distância mostra uma compreensão clara da importância da padronização nesta área, e como isto é um pré-requisito para seu sucesso (DEVEDŽIĆ; JAVONOVIĆ; GAŠEVIĆ, 2007).

Os padrões de metadados são aqueles utilizados para descrever os atributos dos OA, como autores, título ou linguagens, para regular o empacotamento dos OA, para descrever as complexas UoL com vários OA, pacotes, textos e outros recursos *on-line*, para definir a estrutura dos repositórios de OA, para descrever as características do estudante, para especificar os padrões de comunicação, para relacionar as características referentes a qualidade dos OA, entre outros.

Estes padrões permitem a troca de informações entre vários sistemas educacionais usados no processo de aprendizado, permitindo, por exemplo, aos estudantes acessarem conteúdos educacionais de acordo com seu perfil, bem como consultar sua avaliação ou colaborar com outros estudantes na execução de uma UoL.

Metadados permitem aos sistemas e aplicações acessarem os recursos educacionais sem necessidade de interação com o recurso em si (NILSSON, 2008).

Comunidades especializadas em Educação têm definido esquemas próprios de metadados para padronizar a descrição dos recursos educacionais, tais como, *IEEE-LTSC*³⁴, *Dublin Core*³⁵, *ARIADNE (Alliance of Remote Instructional Authoring and Distribution Networks for Europe)*³⁶, *IMS*³⁷, *AICC (Aviation Industry CBT (Computer-Based Training) Committee)*³⁸ e *US DoD (Department of Defense) com ADL (Advanced Distributed Learning)*³⁹, *OKI (Open Knowledge Initiative)*⁴⁰, entre outros. O Quadro 12 apresenta alguns metadados destas comunidades.

³⁴ <http://ieeeltsc.org/>

³⁵ <http://dublincore.org/>

³⁶ <http://www.ariadne-eu.org/>

³⁷ <http://imsproject.org/>

³⁸ <http://www.aicc.org/>

³⁹ <http://www.adlnet.gov/>

⁴⁰ <http://www.okiproject.org/>

Comunidade	Metadado / Padrão
IEEE-LTSC	P1484-12-1: LOM – <i>Learning Object Metadata</i>
	PAPI - <i>Personal and Private Information Standard</i>
ISO/IEC	JTC1 SC36 - <i>Information Technology for Learning, Education, and Training</i>
	<i>Moving Picture Experts Group (MPEG)</i>
Dublin Core	DCMI - <i>Dublin Core Metadata Initiative</i>
ARIADNE	ARIADNE <i>Educational Metadata Recommendation</i>
IMS	IMS LD - <i>Learning Design</i>
	IMS LIP - <i>Learner Information Package</i>
	IMS QTI – <i>Question & Test Interoperability</i>
	IMS CP - <i>Content Packaging</i>
US-Dod-ADL	SCORM - <i>Sharable Content Object Reference Model</i>
AICC	AGRs – <i>AICC Guidelines and Recommendations</i>

Quadro 12– Comunidades e metadados propostos

O projeto e desenvolvimento de especificações de metadados padronizados é global e conta com muitas iniciativas. Iniciativas de fornecedores de sistemas educacionais têm no IMS seu consórcio mais conhecido que conta com participantes de empresas, de universidades e de institutos. O IMS é aberto, mas requer de seus membros uma subscrição anual. Entre seus membros incluem-se o *BlackBoard*, o WebCT, a IBM, OUNL e outros. Iniciativas baseadas no consenso de especialistas incluem IEEE LTSC, ADL e Prometeus (*PROMoting Multimedia access to Education and Training in the European Society*). Outras iniciativas são nacionais ou regionais como ANSI (*American National Standards Institute*) nos Estados Unidos e o CEN (*European Committee for Standardization*) na Europa (HUMMEL et al., 2004).

Os padrões apresentados podem ser estendidos no esquema definido. Além disso, dois ou mais padrões podem produzir a mesma ação. Investigadores, colaboradores, praticantes, e fornecedores algumas vezes usam o melhor de cada padrão para especificar seu próprio padrão. Geralmente, propõem estender alguns elementos do padrão ou combinam diferentes padrões existentes. Leves desvios na rigidez dos padrões e nas especificações em desenvolvimentos práticos podem adicionar valor à usabilidade de um sistema. Por exemplo, o modelo do estudante do projeto Tangram⁴¹ utilizou elementos do estudante do IEEE PAPI (elementos de desempenho) e especificações do IMS LIP (elementos para representar preferências). Isto permite ao sistema desenvolvido interoperar e trocar dados dos estudantes com outras aplicações.

⁴¹ <http://iis.fon.bg.ac.yu/TANGRAM/>

A criação de um padrão internacional requer um esforço complexo no ensino e aprendizagem (BERGGREN et al., 2005). Patrocinadores educacionais têm interesse em um método comum para trocas entre as interfaces, linguagens, códigos e métodos.

Um dos padrões amplamente utilizado e implementado é o IMS LD - trata-se de um padrão para transformar planos de lições regulares em UoL interoperáveis. Esta especificação é feita para usar um modelo pedagógico para obter uma UoL editável e executável, de forma interoperável. Possui muitos elementos como papéis, atividades, ambientes ou recursos e funcionalidades como propriedades, condições, monitoração, serviços ou notificação. O Anexo C aborda o assunto. Ferramentas de edição de PA geram definições para execução em um LMS ou noutro sistema educacional (BURGOS et al., 2007).

Embora cada especificação seja projetada para aumentar a interoperabilidade entre os sistemas, observa-se que cada vez mais sistemas precisam trabalhar com mais de uma destas especificações e isto apresenta um aumento significativo na sua complexidade. A solução é harmonizar os metadados – interoperabilidade entre metadados padrão (NILSSON, 2008) e isto pode ser obtido com o uso de ontologias. Muitos pesquisadores usam ontologias para descrever o domínio de conhecimento e os metadados. Isto tem sido um método poderoso para realçar a força pedagógica dos ambientes de aprendizagem com a descrição dos seus recursos. Uma iniciativa neste sentido é *Resource Aggregation Model for Learning, Education and Training* (RAMLET) do IEEE LTSC⁴² - definição da nomenclatura e do modelo conceitual para recursos digitais para aprendizagem, educação e treinamento.

Os pesquisadores de OA buscam a disseminação e reuso do conhecimento. E investigam como realizar isto efetivamente. WS e ontologias podem trazer uma resposta para esta necessidade, servir como estrutura para indexar o conteúdo dos documentos a partir de diferentes pontos de vista e ajudar na sua formalização e já estão presentes em muitas comunidades de ensino a distância. Muitos dos projetos de pesquisa com OA focam no uso de metadados padrão para aumentar o reuso (ZOUAQ; NKAMBOU; FRASSON, 2007; NILSSON, 2008).

Tecnicamente, a implementação e o uso prático dos padrões estão relacionados à anotação dos OA, mas quando a anotação é realizada manualmente, usam-se metadados prescritos, fixos e, freqüentemente, não apropriados. Ao contrário, a

⁴² <http://www.ieeeltsc.org/working-groups/wg11CMI/ramlet/>

anotação baseada em ontologia pode trazer maior flexibilidade e dinamismo ao associar metadados com OA, fornecendo oportunidades para preenchimento dos metadados com conteúdo relevante, e permitindo a combinação dos metadados de múltiplas origens. Estas anotações baseadas em ontologia servem como ponto de integração semântica de múltiplos LORs.

A estrutura ontológica pode fornecer realimentação para autores de conteúdos sobre uso dos conteúdos, durante o processo de aprendizagem, gerar retorno para os instrutores sobre as atividades, o desempenho e a colaboração dos estudantes (NAJJAR et al., 2005). Também pode personalizar o processo de aprendizagem e reconhecer as peculiaridades e similaridades da situação de aprendizagem (para inferir os OA mais apropriados, presentes no repositório, para aquele contexto) além de promover funcionalidades como suporte aos professores na construção de seus cursos e personalizar rotas de aprendizagem para os estudantes.

As ontologias são introduzidas em muitos projetos para explicitar a conceitualização das experiências de aprendizagem e o modelo de conhecimento do domínio. Isso foi realizado na arquitetura denominada SWALEM (*Semantic Web Assisted Learning Experience Management*), inspirada no modelo de informação do IMS LD, e com uma camada para ontologia. As ferramentas de anotação semântica disponibilizadas para os estudantes permitem anotar seu perfil, as atividades relevantes de aprendizagem e os recursos de aprendizagem do ambiente de aprendizagem. O conhecimento acumulado na forma destas anotações semânticas pode ser reutilizado nas consultas e inferências, de tal forma a realçar e refletir a aprendizagem (TAO et al., 2006).

Para reutilizar OA é necessário um modelo global baseado não somente na descrição deles, mas também na definição dos objetivos de aprendizagem. Descobrir e reutilizar OA são atividades baseadas na disponibilidade de metadados ou anotações semânticas, criados por humanos (MALAXA; DOUGLAS, 2005).

A ação ontológica também é fundamental para a comunicação dos agentes de software. Ela define os conceitos necessários para a troca de mensagens, como uma ontologia tem diferentes níveis de abstração, o conteúdo da mensagem pode ter diferentes dimensões. Ele pode ser geral ou específico. Por exemplo, na comunicação com o agente do modelo de estudante, outro agente pode responder sobre o estado do conhecimento geral do estudante e o seu conhecimento sobre um tópico específico (CHEN; MIZOGUSHI, 1999).

Dicheva, Sosnovsky e Brusilovsky (2005) estruturaram o tema Ontologia para Educação em dois grandes blocos: usando e construindo ontologias para Educação. O Quadro 13 apresenta os aspectos principais desse estudo. Esta organização dá uma visão rápida e abrangente dos temas envolvidos no estudo da WSE. No uso de ontologias na Educação são apresentadas as perspectivas tecnológica e da aplicação. Na primeira destaca-se a recuperação do conhecimento e da informação e a tecnologia para WS como anotação semântica, arquitetura e serviços, compartilhamento, reuso e interoperabilidade de conhecimento. Na segunda aparecem as ferramentas cognitivas para construção e comunicação do conhecimento. Na construção de ontologias estruturam as ações teóricas na Engenharia de Ontologia desenvolvimento de ontologias em geração e extração, construção manual ou colaborativa e suporte existente.

Usando ontologias na Educação	Construindo ontologias para Educação
<p><u>Perspectiva tecnológica</u></p> <ul style="list-style-type: none"> ☐ Tecnologia p/ recuperação de conhecimento: Organização do conhecimento, Inferência do conhecimento e Planejamento ☐ Tecnologia para recuperação de informação: Visualização, Navegação e Consultas ☐ Tecnologia para WS Anotação semântica para: Objetos de aprendizagem, Atividades instrucionais e Atividades de autoria Arquiteturas e serviços dirigidos pela ontologia Compartilhamento de conhecimento Reuso e interoperabilidade de conhecimento <p><u>Perspectiva da aplicação</u></p> <ul style="list-style-type: none"> ☐ Ferramentas cognitivas: Construção do conhecimento, Comunicação, Resultado, e-portfólio e Tipo de conhecimento 	<p><u>Ações teóricas na Engenharia de Ontologia</u></p> <ul style="list-style-type: none"> ☐ Elucidação do conhecimento ☐ Estruturação do conhecimento ☐ Formalização do conhecimento ☐ Mapeamento e junção de ontologias ☐ Evolução de ontologias ☐ Modificação e versionamento de ontologias ☐ Desenvolvimento de ontologias <ul style="list-style-type: none"> ☐ Geração e extração de ontologias: Aspectos de aprendizagem de máquinas, Aspectos recuperação de informação e Aspectos específicos de domínios formal e semi-formal ☐ Desenvolvimento manual de ontologias: Projeto de conceitos, relacionamentos e regras; Restrições e ações pedagógicas ☐ Desenvolvimento de ontologias colaborativas ☐ Suporte ao desenvolvimento de ontologias: Padrões e linguagens e Ferramentas (editores de ontologia e ambientes para engenharia de ontologias)

Quadro 13 - Estrutura para Ontologia na Educação, adaptado de (DICHEVA; SOSNOVSKY; BRUSILOVSKY, 2005)

Alguns esforços iniciais para apoiar estas características semânticas em aplicações WSE são promissores. Por exemplo, o IEEE já aborda padrão para ontologias na

Educação no grupo denominado IEEE *Standard Upper Ontology* (SUO) e desenvolveu um modelo abstrato comum para IEEE LOM em RDFS; o núcleo de Dublin⁴³ e o W3C desenvolveram, no projeto SKOS (*Simple Knowledge Organization Systems*)⁴⁴, especificações para permitir a representação do conteúdo e a estrutura de vários esquemas de conteúdo (como enciclopédias, taxonomias, terminologias e glossários) em RDF.

Devedžić (2006) relacionou as ações e conceitos importantes para a WSE. A Figura 8 destaca os elementos principais: Conteúdo Educacional, Ontologias, Agentes Pedagógicos, Personalização, Linguagens, Tecnologias, Ferramentas e Serviços.

Figura 8 – Blocos Importantes da WSE, adaptada de (DEVEDŽIĆ, 2006)

Os autores de recursos educacionais preparam o Conteúdo Educacional na forma de OA multimídia, exemplos, questões, exercícios, simulações entre outros. O conteúdo é estruturado em UoL coerentes como lições, capítulos, ou testes, baseados em objetivos e propósitos pedagógicos. Ontologias representam o conhecimento essencial e garantem o compartilhamento e reuso do Conteúdo Educacional e a interoperabilidade entre os diferentes sistemas educacionais. Também garantem o registro, a descoberta, a invocação, a composição e monitoração de serviços inteligentes educacionais na WS.

Os Agentes Pedagógicos auxiliam na localização, listagem, seleção, agrupamento, integração e outras tarefas relacionadas com o conteúdo educacional de diferentes servidores educacionais.

Os estudantes estão interessados na Personalização de suas experiências de aprendizagem, através da definição do seu jeito de aprender - estilo, objetivos, preferências e velocidade.

⁴³ <http://dublincore.org/educationwiki/DCMIIEEEELTSCTaskforce>

⁴⁴ <http://www.w3.org/TR/2007/WD-skos-ucr-20070516/>

Diferentes Linguagens são usadas para codificar e apresentar as informações contidas no material de aprendizagem. Algumas linguagens formais podem ser usadas para o desenvolvimento de Conteúdo Educacional, representação de Ontologias e Serviços educacionais, bem como as linguagens de comunicação dos diferentes Agentes Pedagógicos.

Os serviços Web são usados para oferecer aos professores, estudantes e autores Serviços para acessarem o Conteúdo Educacional em um domínio específico de interesse.

Alguns projetos relacionados à Educação com uso de ontologias são:

- RichODL⁴⁵, um ambiente de aprendizado na Web desenvolvido para treinar estudantes na modelagem e simulação de ambientes dinâmicos (ZDRAHAL et al., 2000);
- *ScholOnto Scholarly Ontologies*⁴⁶, uma biblioteca digital baseada em ontologias que possibilita interpretar domínios (SHUM; MOTTA; DOMINGUE, 2000);
- *Automatic Metadata Generation*⁴⁷ (AMG) providencia um sistema para geração automática de metadados na forma de serviços Web e no formato IEEE LOM, com a análise do conteúdo do recurso (CARDINAEELS; MEIRE; DUVAL, 2005);
- Projeto Tangram é um ambiente integrado para aprendizagem no domínio Sistemas Inteligentes de Informação, que usa uma abordagem baseada em ontologia para, automaticamente, anotar OA no ALOCOM (*Abstract Learning Object Content Model*) (JOVANOVIĆ et al., 2007);
- Projeto SeLeNe⁴⁸ (*Self e-Learning Networks*), destinado a transformar documentos DocBook em OA com metadado semântico, representados em seqüências RDF e com relações semânticas (KEENOY; LEVENE; PETERSON, 2004);
- Projeto *Trial-Solution*⁴⁹, no domínio de estudantes universitários matemáticos, ajudando no particionamento de livros eletrônicos existentes para recursos de aprendizagem elementares; utiliza documentos em LaTeX ou Word, e transforma-os em unidades semânticas (BUFFA et al., 2005);
- Projeto IMAT (*Integrating. Materials and Training*), para reuso de manuais técnicos como material de aprendizagem usando análise automática do

⁴⁵ <http://virtual.cvut.cz/odl/>

⁴⁶ <http://kmi.open.ac.uk/projects/scholonto/software.html>

⁴⁷ http://www.cs.kuleuven.be/~hmdb/joomla/index.php?option=com_content&task=view&id=29&Itemid=57

⁴⁸ <http://www.dcs.bbk.ac.uk/~ap/projects/selene/>

⁴⁹ <http://www.uni-koblenz.de/ag-ki/PROJECTS.OLD/TrialSolution/KMS/www/about.html>

documento (documentos PDF) em combinação com técnicas de indexação com ontologias (DE HOOG et al., 2002);

- Arquitetura TELOS (*TelELearning Operating System*), um conjunto de funcionalidades coordenadas e sincronizadas para apoiar interações de pessoas e máquinas nos sistemas de gerenciamento de conhecimento e aprendizagem. Integra agentes humanos e computacionais usando dois processos básicos: representação semântica dos recursos e agregação dos recursos (PAQUETTE, 2007).

Já a ontologia proposta por Verbet e Duval (2004) visa garantir uma descrição formal para a estrutura do conteúdo dos OA. Os autores realizam uma implementação com as ferramentas abertas⁵⁰ e com a ferramenta Microsoft PowerPoint decompondo as apresentações em segmentos - slides, parágrafos, listas, listas de itens, imagens, diagramas e tabelas. Todos os segmentos são categorizados no conteúdo do objeto, usando modelos padronizados para texto e anotados no *framework* AMG. O processo de agregação é usado quando os autores querem construir novos OA.

Amorim, Lama e Vila (2006) propuseram o desenvolvimento de uma taxonomia de conceitos com base na especificação IMS LD para descrever, explícita e formalmente, a semântica de seus elementos, presentes no modelo conceitual e um conjunto de axiomas para representar as restrições semânticas, contidas nas explanações informais dos modelos de informação e comportamental. Esta ontologia é atualmente utilizada pelo LMS desenvolvido no projeto EUME (*Ubiquitous and Multimedia Enviroment for Education*) e conta com aproximadamente 50 axiomas definidos no formato OWL. O projeto EUME suporta atividades de ensino, tais como gerenciamento de autoria dos recursos, aprendizagem colaborativa, educação a distância e tutorial personalizado.

Mizoguchi, Hayashi e Bourdeau (2007) elaboraram o Smarties, um protótipo de uma ferramenta de autoria baseada numa ontologia para construção de cenários de aprendizagem. A ontologia foi construída com o editor Hozo e chama-se Omnibus. O resultado pode ser representado no padrão IMS LD.

O uso de padrões reconhecidos por comunidades conceituadas aliado à construção da ontologia correspondente torna-se um marco poderoso na busca pela interoperabilidade, facilidade de acesso e reuso dos recursos educacionais como OA e PAs na Educação. Todos os envolvidos compartilham do mesmo entendimento

⁵⁰ <http://www.openoffice.org/>

semântico sobre os recursos e podem decidir sobre seu melhor aproveitamento. Porém, muito mais do que especificações de metadados e ontologias, é necessário ter recursos com seus metadados associados.

3.3 Sistemas Educacionais

Koper (2001) construiu um Modelo de Referência Arquitetural, que pode ser visto na Figura 9. Esta arquitetura orienta a distribuição dos diferentes sistemas educacionais existentes nas instituições de ensino e aprendizagem em três grandes blocos Desenvolvimento, Publicação e Entrega. O modelo segue a notação padrão UML. Assim, por exemplo, a associação entre os componentes de Autoria e Desenvolvimento é uma relação de generalização, e as linhas pontilhadas representam relacionamentos de dependência.

Figura 9 – Modelo de Referência Arquitetural, adaptada de (DEVEDŽIĆ, 2006)

O desenvolvimento de conteúdos educacionais demanda habilidades multidisciplinares dos projetistas envolvidos e isto pode levar alguns professores para o papel de projetista durante sua realização permitindo assim a entrega do conteúdo construído para um número maior de professores.

3.3.1 Desenvolvimento

Projetistas trabalham em ambientes de desenvolvimento (Autoria, Projeto e Gerenciamento de Conteúdo) e tipicamente criam e/ou alteram OA e PAs. Ambientes de autoria requerem ontologia, domínio, pedagogia e anotação. Tanto os OA quanto os PAs precisam ser anotados para facilitar a busca, a localização e sua filtragem. Desenvolvedores usam um projeto instrucional padronizado para descrição dos OA e UoLs. Ferramentas de Gerenciamento de Conteúdo são usadas para estruturar os OA e UoLs criados.

Kaskalis, Tzidamis e Margaritis (2007) realizaram uma pesquisa com quarenta e quatro (44) ferramentas de autoria orientadas a recursos multimídia para Educação, observando requisitos como: facilidade de uso, produtividade, interação técnica e preço, entre outros. Atribuíram notas para cada requisito, e no final, sete ferramentas ficaram entre as melhores pontuadas: Opus Pro, ToolBook Instructor, Director, Flash, Authorware, Liquid Media e Mediator. Os autores observaram as ferramentas sob o ponto de vista educacional e concluíram que há falta de requisitos especiais para o ambiente educacional.

Wang (2006) relacionou e detalhou as ferramentas de autoria baseadas no padrão SCORM mais citadas nos artigos do IEEE e ACM (*Association for Computing Machinery*) entre elas: Reload (*Reusable eLearning Object Authoring & Delivery*), Manifest Maker, Thesis, *Metadata Generator Lite*, *Course Creation Toolkit*, VOSSAT (*Visualized Online Simple Sequencing Authoring Tool*) e MAGIC (*Metadata Automated Generation for Instrucional Content*).

Dentre as ferramentas para anotação citadas no primeiro capítulo destacam-se para área educacional: Annotea, SHOE, COHSE, AeroDAML, Melita e OntoMat-Annotizer.

Exemplos das ferramentas para edição de PAs no padrão IMS LD, com seus níveis (A, B, C) são encontrados no Anexo C.

3.3.2 Publicação

As ferramentas para Publicação garantem a disponibilidade na Web do material criado pelos desenvolvedores. Publicação pode ser na forma estática (não

adaptativa), apoiar a entrega de OA na forma de páginas HTML estáticas, e-livros ou CDs. Porém pode entregar na forma dinâmica (adaptativa), de acordo com o modelo do estudante ou serviços Web.

3.3.3 Entrega

A Entrega ocorre com LMS, LOR, os Servidores Educacionais e outros sistemas educacionais.

Os LMSs providenciam um conjunto de funcionalidades para apoiar as atividades de aprendizagem dos estudantes como: armazenar arquivos, realizar fóruns, estabelecer calendário, sugerir novidades, propiciar serviço de e-mail, coordenar grupos de pesquisa, fornecer organização, possibilitar pesquisas nas perguntas e respostas mais freqüentes ou publicar cronogramas. Também, disponibilizam funcionalidades para os professores e para os profissionais administrativos. Podem ser agrupados em duas categorias: iniciativas *Open Source*⁵¹ como Moodle⁵², Sakai⁵³, ATutor⁵⁴ e Whiteboard⁵⁵ e soluções proprietárias como WebCT/Blackboard⁵⁶, Gradepoint⁵⁷, Desire2Learn⁵⁸ e Learn.com⁵⁹ (DAGGER et al., 2007). No Brasil alguns exemplos são CoL⁶⁰, TIDIA-Ae, Teleduc⁶¹, AulaNet⁶² e e-Proinfo⁶³.

Além desta classificação, os LMS podem ser classificados em gerações conforme os padrões e tecnologias utilizadas, o que pode ser visto na Figura 10.

⁵¹ <http://www.edtechpost.ca/pmwiki/pmwiki.php/EdTechPost/OpenSourceCourseManagementSystems>

⁵² www.moodle.org

⁵³ www.sakaiproject.org

⁵⁴ www.atutor.ca

⁵⁵ <http://whiteboard.sourceforge.net>

⁵⁶ www.blackboard.com

⁵⁷ www.gradepoint.net

⁵⁸ www.desire2learn.com

⁵⁹ www.learn.com

⁶⁰ ⁶⁰ <http://col.usp.br/portal/>

⁶¹ <http://teleduc.nied.unicamp.br/>

⁶² http://www.eduWeb.com.br/portugues/elearning_tecnologia.asp

⁶³ <http://www.eproinfo.mec.gov.br/>

Figura 10 – Classificação dos LMS em gerações, adaptada de (DAGGER et al., 2007)

Kljun et al. (2007) realizaram um estudo de artigos escritos em diferentes épocas, disponíveis, citados e muito procurados na Web, sobre classificação ou avaliação de ferramentas LMS. Estabeleceram um padrão para os critérios comparados nos artigos, separados em grupos sobre conteúdo, comunicação, ambiente, colaboração, educação, gerenciamento e técnico. Este estudo fortalece a necessidade dos LMS tratarem o padrão SCORM para importar, exportar e editar seus conteúdos tendo em vista a portabilidade e reutilização dos recursos tão essenciais para a colaboração, o trabalho em grupo, formação de redes sociais e apoio adicional ao ensino-aprendizagem.

Em 2005, MacLeod identificou os repositórios considerados mais importantes e além de avaliar outras características como atendimento aos padrões de metadados, citou a quantidade de objetos públicos disponíveis em cada repositório. O Quadro 14 dá uma visão dos resultados desta pesquisa.

Repositório	Sigla /Origem	Website	Objetos Públicos
ADLib	Athabasca University Digital Library Universidade Athabasca	adlib.athabascau.ca	500
ARIADNE	Fundação ARIADNE	www.ariadne-eu.org	2526
CAREO/Apollo	Campus Alberta Repository of Educational Objects Universidade Calgary	apollo.ucalgary.ca	3600
CLOE	Co-Operative Learning Object Exchange Universidade Waterloo	www.cloe.on.ca	50
EdNA	Education Network Australia Education.au	edna.edu.au	30300
eRIB	eduSource Repository in a Box Centro de Pesquisa da Tele-universidade de Quebec	www.edusource.ca	49761
MERLOT	Multimedia Educational Resource for Learning and Online Teaching	www.merlot.org	106707
NIME	National Institute of Multimedia Education NIME	www.nime.ac.jp	100
POOL	Portal for Online Objects in Learning Universidade Simon Fraser	www.eduplash.net	3000
TILE	The Inclusive Learning Exchange Universidade de Toronto	inclusivelearning.ca	2371

Quadro 14 - Principais LORs (MACLEOD, 2005)

Outros repositórios de OA são PILO (*Practitioner Inquiry Learning Object*) para colecionar recursos multimídia para professores, CeLeBraTe (*Context e-Learning with Broadband Technologies*) para compartilhar e gravar recursos de aprendizagem, SMETE⁶⁴ (*Science Mathematics Engineering Technology Education*) que suporta buscas e visualização de recursos no domínio da ciência, matemática, engenharia e tecnologia. A coleção tem uma grande variedade de tipos, padrões de metadados, protocolos, esquemas de autenticação e modelos de negócio (HATALA et al., 2004). Resulta do projeto NSDL (*National Science Digital Library*) fundado pela *National Science Foundation*.

LGR (*Lydia Global Repository*) é LOR comercial onde os usuários, após realizarem seu registro, procuram os OA. Os metadados dos OA descrevem o proprietário e o valor, e os usuários podem então adquiri-los, via transação comercial no site, e acessá-los no VPR (*Virtual Private Repository*).

ULTRA (*UREKA Learning-Object Taxonomy & Repository Architecture*) é um repositório para gravar, compartilhar, reutilizar e classificar OA, *digital assets* (como filmes, animações, imagens, etc.) e outros conteúdos. Tem duas partes definidas como SMU (*Semantically Meaningful Unit*) e CMF (*Common Media Framework*) (IHSAN et al., 2006).

⁶⁴ <http://www.smete.org/smete/>

A criação de arquiteturas contemplando Serviços Educacionais é um tema recente. São exemplos o projeto Elena/Edutella⁶⁵, o projeto ONES (*One-Stop eLearning Portal*), o projeto MIWeb⁶⁶ (*Mediator-based Integration of Web Sources*) e o projeto ROSA (*Repository of Objects with Semantic Access*).

O projeto Elena/Edutella provê metadados baseados em RDF em uma infraestrutura P2P para a troca de recursos educacionais, usando padrões para OA como IEEE LOM, IMS e ADL SCORM. Consiste de serviços de consulta, anotação, registro e mediação.

O projeto ONES tem como objetivo desenvolver um sistema para apoiar a busca por cursos de interesse do estudante em universidades virtuais. Devido ao fato de existir heterogeneidade entre as universidades virtuais, o sistema ONES provê mecanismos para esconder a distribuição dos portais de *e-learning*.

MIWeb é um sistema que contribui com uma arquitetura de mediador no contexto de fontes de dados na Web para busca de materiais de aprendizagem e publicações. O sistema MIWeb pesquisa em fontes de dados que descrevem tipos diferentes de documentos Web, tais como, a máquina de busca Google, a fonte científica Citeseer e recursos específicos para *e-learning* desenvolvidos no projeto NewEconomy. MIWeb utiliza padrões de metadados para integração e tecnologias como HTTP, RDF e a linguagem de consulta RDQL. A heterogeneidade semântica entre as informações é tratada com a utilização de RDFS e OWL.

ROSA é um repositório de OA, desenvolvido pelo Instituto Militar de Engenharia do Rio de Janeiro, com acesso semântico a partir das linguagens OWL e SWRL. Na definição de sua arquitetura contempla o padrão MOF.

O domínio da Educação possui um leque abrangente de sistemas educacionais, desde os mais conhecidos para entrega como os LMS até os associados à autoria dos recursos. Uma instituição de ensino possui vários destes sistemas e a interoperabilidade entre eles torna-se crucial, não somente sob a ótica técnica como também na pedagógica. E esta mesma instituição pode compartilhar seu repositório ou serviços com outras instituições. A arquitetura da WS com seus diversos tipos de ontologias promove a interoperabilidade e o reuso principalmente ao construir ontologias baseadas em metadados padronizados.

⁶⁵ <https://edutella.dev.java.net/>

⁶⁶ <http://bird.cs.tu-berlin.de:8000/mediator/index.html>

3.4 Discussão

O domínio da Educação está em constante evolução impulsionado tanto pelas mudanças na demanda social com a aprendizagem presente ao longo de toda a vida pessoal e profissional do indivíduo, como pelas mudanças no processo de ensino e aprendizagem, com as exigências para maior flexibilidade e adaptabilidade ao perfil dos estudantes, como também pelas mudanças organizacionais nas instituições educacionais, com a descentralização e inexistência de limites físicos. Este cenário leva ao uso de novas tecnologias, de acordo com o tipo de aprendizagem escolhido, como multimídia e ferramentas de trabalho cooperativo, e também considerando que a aprendizagem ocorre sem que nenhuma pessoa deixe seu posto de trabalho, de estudo ou sua moradia. Tudo isto sem esquecer o objetivo maior da Educação que é permitir a cada pessoa resolver os problemas do cotidiano, compreender o mundo e atuar na transformação de seu contexto.

As instituições educacionais têm disponível diversos sistemas educacionais para desenvolvimento, publicação e entrega de OA, UoL ou Serviços Educacionais. Esta diversidade leva comunidades a definirem metadados padronizados para, principalmente, facilitar a interoperabilidade entre os sistemas. Porém, os padrões atuais não incluem formalmente a semântica descrita; assim, devem evoluir para tratar mecanismos de raciocínio e busca semântica baseados em ontologias. Estas combinam conceitos sobre conhecimento, habilidades, atitudes e avaliação, necessários no ambiente educacional, definem competências individuais dos atores, pré-requisitos e objetivos para atividades e conteúdo dos recursos, critérios de avaliação e personalização das capacidades para a aprendizagem.

Usuários podem ter ajuda na busca por informações relevantes com aplicação de inferências e cálculos, e na combinação da informação em novas formas. Também um estudante pode esperar por um instrutor virtual para estimular sua aprendizagem, como um educador faz, no mundo real. O outro lado da moeda é que os autores de conteúdo devem fornecer o significado do domínio específico e explícito para permitir a interpretação automática pelas máquinas.

A introdução destas inovações tecnológicas em grande escala depende de mudanças na concepção e organização dos recursos educacionais, nos modelos pedagógicos e na forma de pensar dos envolvidos – administradores, educadores, professores, estudantes, monitores etc. - e de fortes investimentos em tecnologia.

4 **FRAMEWORK PARA APRENDIZADO DE ONTOLOGIAS**

Os engenheiros de ontologias têm a difícil tarefa de construir, rápida e corretamente, muitas ontologias em diferentes domínios, integradas e representando um consenso entre os especialistas do domínio. Tudo isso é necessário para a WS cumprir seu objetivo: compartilhar o conhecimento entre máquinas e humanos. Apesar da Engenharia de Ontologias apresentar uma grande diversidade de técnicas, métodos e ferramentas para auxiliá-lo, nem todos estão totalmente padronizados, divulgados e conhecidos. Aliado a isso, entretanto, existe muito conhecimento representado em sistemas existentes, pronto para ser aproveitado no universo da WS através de suas ontologias.

O *framework* proposto visa unir a Engenharia de Software e a Engenharia de Ontologias, através do uso dos padrões OMG e W3C, para construção acelerada de representações iniciais de ontologias, a partir de recursos dos sistemas existentes, mais especificamente textos e *schematas*.

Os padrões OMG – MDA, UML, ODM, MOF e XMI – inserem o uso de meta-metamodelo nas ferramentas utilizadas no ambiente de desenvolvimento de software, principalmente, para permitir a interoperabilidade entre seus metamodelos. W3C e sua arquitetura da WS, centralizada na OWL, possibilita que os conteúdos dos recursos Web possam ser consultados, deduzidos, inferidos, enfim compreendidos, pelas máquinas.

A construção acelerada ocorrerá pela diminuição do tempo do engenheiro de ontologia na compreensão do domínio junto ao especialista e na elaboração das representações iniciais sintaticamente corretas.

A presença de metamodelos é um aspecto importante do *framework* proposto. O metamodelo de uma ferramenta utilizada contém modelos reais, implementados por ela. Os editores de modelos, como uma ferramenta qualquer para edição de ontologia, são usados para definição e manutenção destes modelos, bem como seu armazenamento num repositório e visualização. Todos os mecanismos e algoritmos usados para avaliação e uso dos modelos são realizados de acordo com as primitivas presentes no metamodelo. No momento que estes modelos precisam ser inseridos em outra ferramenta, como em uma ferramenta para integração de

ontologias, os processos de transformação entre os metamodelos das ferramentas garantem a expressividade dos elementos constantes do modelo.

As premissas consideradas na construção do *framework* foram:

- o engenheiro de ontologia desconhece ou conhece pouco sobre o domínio a ser representado na ontologia;
- a constante evolução e surgimento de distintas ferramentas para AO a partir de diferentes fontes, aplicando várias técnicas combinadas;
- a engenharia progressiva da ontologia será sempre necessária;
- as ferramentas para participarem do *framework* possuem ou permitem adequação de seu metamodelo ao padrão proposto.

Para melhor detalhar o proposto seguem sua definição formal, o modelo OWL, a correspondência entre os metamodelos, a visão geral dos blocos do *framework*, os requisitos das ferramentas e suas atividades a serem inseridas na Engenharia de Ontologias.

4.1 Definição Formal

Dado um conjunto de recursos representativos do sistema existente, aplica-se um conjunto de técnicas para aprendizado dos elementos da ontologia, obtendo-se um conjunto de representações iniciais, tratado num ambiente de ontologias e disponível para uso nas atividades da engenharia progressiva.

Os recursos representativos considerados são principalmente os *schematas* e os textos. Os *schematas* estão na forma de modelo de dados ou diagramas de classes da UML construídos para o sistema existente. Os textos estão na documentação do sistema existente na forma de tutoriais, padrões de domínio e documentos diversos sobre o sistema existente.

O conjunto de técnicas para AO encontra sua implementação em diferentes ferramentas ou ambientes (dada a diversidade de abordagens vistas em 2.6.3 e resultados atualizados de pesquisas). Algumas ferramentas apóiam seus modelos em padrões para metamodelos, como as ferramentas CASE com os diagramas UML, e outras em definições internas proprietárias, como as ferramentas para AO a partir de textos.

Neste ponto, o *framework* proposto introduz a utilização do ODM e seus metamodelos já definidos para representar a OWL, a UML versão 2 (UML2) e os modelos de dados. ODM é o metamodelo que permite juntar os padrões do OMG (MDA) e do W3C (OWL) seguindo o meta-metamodelo padrão definido pelo MOF. MOF é a base da MDA e de muitos outros metamodelos da MDA tornando possível derivar modelos para *OWL Lite*, *DL* ou *Full* e verificar a consistência dos modelos obtidos. O conjunto de representações iniciais segue o ODM para OWL mesmo para representações intermediárias das ferramentas.

Assim, as diversas ferramentas participantes do *framework* - AO, CASE, Transformação de Metamodelos e Ambiente de Ontologia –, representadas na camada M2 da Figura 11, seguem o padrão MOF, têm seus metamodelos (M2) definidos de acordo com o meta-metamodelo (M3) e tratam diversos modelos (M1).

Figura 11 – Representação dos Metamodelos no *Framework*

Isto garante que todos os modelos (M1) considerados pelas diferentes ferramentas tenham seu metamodelo compartilhado entre as ferramentas e, desta forma, todas as instâncias (M0) dos modelos têm a interoperabilidade garantida entre as ferramentas, através das regras de transformação entre metamodelos aplicadas sobre os arquivos XMI – instâncias de modelos serializados de acordo com o metamodelo.

Por exemplo, uma mesma ferramenta CASE pode ter definido em seu metamodelo os modelos E/R, diagrama de classes, diagrama de seqüência e outros. Cada modelo trata seus elementos de acordo com as primitivas do metamodelo implementadas. No caso do modelo E/R, tratam-se os elementos entidades, relacionamentos e atributos, entre outros, e uma das primitivas mais conhecidas é que toda entidade precisa ter um atributo classificado como identificador único.

Já, ao se considerar o metamodelo de uma ferramenta para AO, pode ocorrer um modelo para cada técnica empregada. Mas os elementos principais dos modelos serão conceitos, sinônimos, hierarquia de conceitos, relações e instâncias. Estas ferramentas podem utilizar modelos intermediários ou temporários requeridos por seus algoritmos ou funcionalidades, e uma das primitivas consideradas é que o algoritmo executado atribui um grau de certeza junto ao elemento aprendido.

O Ambiente de Ontologia também seguirá o ODM para OWL e isto significa que as ferramentas participantes deste ambiente para tratar (editar, integrar, validar, importar, visualizar etc.) os elementos da ontologia – conceitos, relações, instâncias, funções e axiomas - compartilham o mesmo metamodelo.

Muitas ferramentas para Aprendizado de Ontologias, CASE e Ambiente de Ontologias apresentam a funcionalidade para extração das instâncias (M0) de seus modelos (M1), e outra para importar instâncias de modelos conhecidos - já implementou as regras de transformação entre metamodelos determinados -, ambas através de arquivos XML. Este arquivo serializa os elementos presentes no metamodelo de origem de tal forma que as instâncias representadas nos modelos possam ser aproveitadas por outras ferramentas com metamodelos distintos. Porém, nem todas ferramentas possuem estas funcionalidades, pois o número de modelos possíveis é alto, as combinações (origem/destino) nem sempre são conhecidas ou consideradas e nem todas ferramentas têm seus metamodelos de acordo com o padrão para meta-metamodelo (M3). Por isso, a Ferramenta para Transformação de Metamodelos faz o mapeamento entre os elementos de cada modelo origem para modelo destino e aplica as regras necessárias para evitar ao máximo a perda de representatividade dos elementos considerados em cada modelo. O modelo destino escolhido foi o modelo padrão OWL representado no ODM.

4.2 Modelo OWL

OWL é um padrão W3C que já foi apresentado na seção 2.4. O Quadro 15 mostra os principais Elementos da Ontologia - como conceito (C), indivíduos (I), propriedades (P), relações definidas em C, funções definidas em C e axiomas - e o elemento correspondente no modelo OWL. A coluna Comentário esclarece a definição adotada pelo modelo OWL para cada elemento. Estes elementos compõem, em todo ou em parte, os modelos das ferramentas do Ambiente de Ontologias e AO, e, por isso, são considerados pela Ferramenta para Transformação de Metamodelos.

Os axiomas são os elementos mais característicos e específicos na representação de uma ontologia e não encontram correspondência na maioria dos recursos representativos dos sistemas existentes, somente no caso dos sistemas que adotaram os princípios da Inteligência Artificial ou usaram recursos da *Object Constraint Language* (OCL) nos diagramas UML, por exemplo.

Elemento da Ontologia	Elemento do Modelo OWL	Comentário
Ontologia	owl:Ontology	Classe raiz
Conceito		
Classe	owl:Class	C e específicas: x, y, z..
Propriedade	owl:OntologyProperty	P
Tipos de dados	owl:DatatypeProperty	entre instâncias da classe
De objetos	owl:ObjectProperty	entre instâncias de duas classe
Relação taxonômica		
Classe	subClassOf	
Propriedade	subPropertyOf	
Função		
Propriedade		
	&owl; FunctionalProperty	se P(x,y) e P(x,z) então y=z
	&owl; SymmetricProperty	se P(x,y) então P(y,x)
	&owl; TransitiveProperty	se P(x,y) e P(y,z) então P(x,z)
	&owl; InverseFunctionalProperty"	se P(x,y) e P(z,y) então x=z
Classe		
	owl:complementOf	C = complementOf(C1)
	owl:disjointWith	C1 != C2
	owl:intersectionOf	C = intersectionOf(C1, C2, ...)
	owl:equivalentClass	C1 = C2
	owl:unionOf	C = unionOf(C1, C2, ...)
Restrição	owl:Restriction	
	owl:cardinality	cardinality(P) = n
	owl:maxCardinality	maxCardinality(P) = n
	owl:minCardinality	minCardinality(P) = n
	owl:allValuesFrom	P(x,y) tem y=allValuesFrom(C)
	owl:someValuesFrom	P(x,y) tem y=someValuesFrom(C)
	owl:hasValue	hasValue: P(x,y) e y=hasValue(I)
Indivíduos	Instances	I
	owl:sameAs	I1 = I2
	owl:differentFrom ...	I1 != I2
	owl:AllDifferent>	I1 != I2, I1 != I3, I2 != I3, ...
	owl:one of	C=I1, I2, I3, ... (enumeração)
	owl:Thing	Conjunto de todos os indivíduos
	owl:Nothing	Conjunto vazio
Axiomas	<i>Axiom</i> ::=	Diversas formas

Quadro 15 - Elementos OWL

4.3 Correspondência entre Metamodelos

Nos últimos anos vários sistemas foram construídos utilizando o modelo de dados e os diagramas UML, principalmente, o diagrama de classes. As ferramentas CASE

para representarem esses modelos, geralmente, seguem o metamodelo padronizado pelo OMG. Isto permite realizar extrações das instâncias dos modelos de seus repositórios em arquivos XMI e aproveitá-los em outras ferramentas existentes no ambiente de desenvolvimento dos sistemas e, também, no Aprendizado de Ontologias.

Nesse *framework*, a transformação de metamodelos considera três grupos de recursos dos sistemas existentes: o modelo de dados, os diagramas UML e os textos.

4.3.1 Modelo de Dados

O modelo de dados é representado em dois modelos: modelo E/R e o Modelo Relacional.

O modelo E/R tem como objetivo modelar conceitualmente as informações representativas do negócio na forma de entidades, atributos, relacionamentos e cardinalidade. Além do dicionário de dados, onde se definem as informações, apresenta uma representação gráfica com símbolos apropriados para cada elemento considerado no modelo. Assim, mais do que uma sintaxe bem definida, possui uma semântica associada a cada elemento.

Já o Modelo Relacional tem como objetivo representar as informações num modelo físico que usa tabelas, colunas, chaves primárias e estrangeiras, linhas e restrições. Este modelo é criado no catálogo ou dicionário de dados do SGBDR.

As ferramentas CASE realizam tarefas tanto para engenharia progressiva – como a transformação do modelo E/R para o Modelo Relacional, incluindo *scripts* SQL para criação dos elementos no SGBDR - como para engenharia reversa – como a leitura do dicionário de dados do SGBDR para construção do modelo E/R.

As abordagens existentes para AO a partir de modelos E/R estão baseadas na análise explícita do esquema relacional (relações, colunas, colunas chaves, dependências funcionais), instâncias e consultas dos usuários. Li, Du e Wang (2005) propuseram o aprendizado automático de ontologias a partir de banco de dados relacionais, expressando o resultado no formato OWL, usando grupos de regras organizadas em cinco grupos: regras para aprendizado de classes, de propriedades,

de hierarquia, de cardinalidade e de instâncias. Por exemplo, uma regra para aprendizado de cardinalidade é assim descrita:

Regra 10 – Para cada relacionamento R_i e $A_i \in \text{attr}(R_i)$, se A_i é declarada como “NOT NULL”, então a *minCardinality* da propriedade P_i correspondente a A_i é 1, A_i é uma coluna identificada como chave estrangeira, R_i é um relacionamento e $\text{attr}(R_i)$ é uma função que retorna os atributos do relacionamento.

Estas regras foram aplicadas nas representações existentes nos sistemas da Universidade Renmin, na China. O resultado do processo foi uma ontologia com vinte classes, quatrocentas propriedades e milhares de instâncias.

Já Trinkunas e Vasilecas (2007) aplicaram mapeamentos e definições de regras sugeridos pelo ODM para construir um modelo de dados a partir de uma ontologia definida em OWL. Os autores utilizaram cinco transformações elementares – o domínio é a ontologia representada (*OWLOntology*), as entidades são as classes da ontologia (*OWLClass*), os atributos são os tipos de dados (*OWLObjectProperty*), os relacionamentos são as propriedades de objeto (*OWLObjectProperty*) e os domínios dos atributos são um tipo de literal ou enumeração de literais (*OWLObjectRange*).

Outros estudos são Sugumaran e Storey (2006), Upadhyaya e Kumar (2005), Li, Du e Wang (2005) e Astrova (2004). Todos estes estudos somados à especificação ODM serviram para estabelecer o mapeamento entre os metamodelos necessário para transformar as instâncias dos modelos E/R e Modelo Relacional em instâncias na representação OWL, como pode ser observado no Quadro 16.

Os elementos da ontologia mapeados nesta transformação são poucos, porém encontram no mapeamento de instâncias, linhas e indivíduos sua maior força. Milhares de dados – instâncias ou linhas - povoam bancos de dados que foram construídos a partir de modelos E/R e Relacional e tê-los disponibilizados, mesmo na versão OWL *Lite*, pode ser muito útil para a WS.

Elemento E/R	Elemento Relacional	Elemento Ontologia	Fragmentos Sintaxe OWL
Entidade	Tabela	Classe	<code><owl:Class rdf:ID=" " /></code>
Atributo	Coluna	Propriedade de tipos de dados	<code><owl:DatatypeProperty rdf:ID=" " /></code> <code><rdfs:domain rdf:resource=" " /></code> <code><rdfs:range rdf:resource=" " /></code> <code></owl:DatatypeProperty></code>
		Propriedade de objeto	<code><owl: ObjectProperty rdf:ID=" " /></code> <code><rdfs:domain ></code> <code><owl:Class></code> <code>owl:unionOf rdf:parse Type=" "></code> <code>owl:Class rdf:about=" " /></code> <code>owl:Class rdf:about=" " /></code> <code><owl:Class></code> <code><rdfs:domain></code> <code><owl:ObjectProperty></code>
Relacionamentos (1:N) hierárquicos	Chave Estrangeira	Relação taxonômica entre classes	<code><owl:Class rdf:ID=" "></code> <code><rdfs:subClassOf rdf:resource=" " /></code> <code><owl:Class></code>
Cardinalidade	Chave Estrangeira	Restrição de cardinalidade	<code><owl:Restriction></code> <code><owl:onProperty rdf:resource=" " /></code> <code><owl:minCardinality">1</owl:minCardinality></code> <code><owl:maxCardinality">1</owl:maxCardinality></code> <code><owl:Restriction></code>
Instância	Linha	Indivíduos	<code><owl:Thing rdf:ID=" " /></code>

Quadro 16 – Correspondência entre Metamodelos E/R, Relacional e OWL

4.3.2 Diagrama de Classes

O diagrama de classes é o principal instrumento da orientação a objetos para representar os conceitos significativos no formato de classes, atributos, métodos, associações e cardinalidade. O formalismo do diagrama pode ser utilizado para representar elementos do negócio ou características de implementação. Assim, existem diagramas de classe do modelo de análise que apresentam elementos significativos do negócio e diagramas de classe do modelo de projeto com elementos significativos pertencentes à visão de arquitetura de software. Os primeiros são mais interessantes para a obtenção dos elementos da ontologia.

As ferramentas CASE representam estes modelos de análise e projeto em seus repositórios de dados na forma de metamodelos padronizados seguindo a MDA.

A transformação entre metamodelos do *framework* proposto utiliza as discussões realizadas na construção do ODM, considerando as limitações da representação

direta dos elementos do modelo UML para os elementos do modelo OWL. Aqui, a ontologia não será modelada com o *profile* UML ou OUP, como proposto pelo ODM, mas será aproveitado o conhecimento já representado em diagramas UML sem OUP para representar uma ontologia em OWL.

Neste caso os diagramas UML não apresentam os elementos introduzidos no OUP do ODM na forma de esteriótipos, marcação e restrições. Os diagramas já existentes possuem as características UML sem equivalência na OWL, como discutido no Anexo A.

Estudos neste assunto encontram-se em Bouchiha, Malki e Benskimane (2007), Cranefield e Pan (2007) e Brockmans et al. (2006).

As correspondências entre elementos dos modelos podem ser observadas no Quadro 17.

O modelo UML é mais rico em suas representações quando comparado ao modelo E/R e permite a obtenção de um número maior de elementos da ontologia.

Elemento UML	Elemento Ontologia	Fragmentos Sintaxe OWL
Classe	Classe	<owl:Class rdf:ID=" " "
Atributo	Propriedade Tipo de Dados	<owl: DatatypeProperty rdf:ID=" " "/> ... </owl:DatatypeProperty>
Associação binária	Propriedade de Objeto	<owl: ObjectProperty rdf:ID=" " "/> ... <owl:ObjectProperty>
Subclasse	Relação taxonômica entre classes	<owl:Class rdf:ID=" "> <rdfs:subClassOf rdf:resource=" " "/> <owl:Class>
Generalização	Relação taxonômica entre propriedades	<owl:ObjectProperty rdf:ID=" "> <rdfs:subPropertyOf rdf:resource=" " "/> <rdfs:range rdf:resource=" " "/> ... </owl:ObjectProperty>
Associação n-árias	Classes ou propriedades	
Associação entre classes	Classes ou propriedades	
Enumeração	Instâncias	<owl:Class rdf:ID=" "> <owl:oneOf rdf:parseType=" "> <owl:Thing rdf:about=" "/> <owl:Thing rdf:about=" "/> </owl:oneOf> </owl:Class>
Disjunção	Função entre classes	<owl:Class rdf:ID=" "> <rdfs:subClassOf rdf:resource=" "/> <owl:disjointWith rdf:resource=" "#"/> <owl:disjointWith rdf:resource=" "#"/> ... </owl:Class>
União	Função entre classes	<owl:Class rdf:ID=" "> <owl:unionOf rdf:parseType=" "> <owl:Class rdf:about=" "#" /> <owl:Class rdf:about=" "#" /> </owl:unionOf> </owl:Class>
Pacote	Ontologia	<owl:Ontology rdf:about=" ">
Dependência	Propriedade	
Multiplicidade	Restrição de cardinalidade	<owl:Restriction> <owl:onProperty rdf:resource=" " "/> <owl:minCardinality">1</owl:minCardinality> <owl:maxCardinality">1</owl:maxCardinality> <owl:Restriction>
Instância	Indivíduo	<owl:Thing rdf:ID=" " " />

Quadro 17 - Correspondência entre Metamodelos UML e OWL

4.3.3 Textos

As várias ferramentas utilizadas para extração de elementos da ontologia representam o conhecimento aprendido num metamodelo instanciado de acordo com as primitivas consideradas para ele. Os resultados parciais, obtidos pela ferramenta nas diversas fases de aprendizado, são independentes da linguagem de representação final da ontologia. Isto deixa a ferramenta flexível e extensível para outras representações. Por outro lado, primitivas precisam ser conhecidas para as regras de transformação entre metamodelos.

O Quadro 18 apresenta os elementos do *Learned Ontology Model* (LOM) proposto por Haase e Volker (2005) para ferramentas de AO a partir de textos.

Elementos	Elemento Ontologia	Fragmentos Sintaxe OWL
Conceito	Conceito (Classe ou Propriedade)	<code><owl:Class rdf:ID="" "</code>
Subconceito de	Relação taxonômica entre classes	<code><owl:Class rdf:ID="" "> <rdfs:subClassOf rdf:resource="" "/> </owl:Class></code>
	Relação taxonômica entre propriedades	<code><owl:ObjectProperty rdf:ID="" "> <rdfs:subPropertyOf rdf:resource="" "/> <rdfs:range rdf:resource="" "/> ... </owl:ObjectProperty></code>
Parte de	Relação taxonômica	
Equivalência	Função entre classes	<code><owl:Class rdf:ID="" "> <owl:equivalentClass rdf:resource="" "> </owl:Class></code>
Igualdade	Função entre instâncias	<code>< ... rdf:ID="" "> <owl:sameAs rdf:resource="" "#"/> </...></code>
Disjunção	Função entre classes	<code><owl:Class rdf:ID="" "> <rdfs:subClassOf rdf:resource="" "/> <owl:disjointWith rdf:resource="" "#"/> <owl:disjointWith rdf:resource="" "#"/> ... </owl:Class></code>
Instância	Indivíduo	<code><owl:Thing rdf:ID="" " "/></code>

Quadro 18 – Correspondência entre Metamodelos LOM e OWL

4.4 Visão Geral

A Figura 12 apresenta os quatro grandes blocos do *framework* proposto: Recursos do Domínio, Aprendizado de Ontologias, Ambiente de Ontologias e Engenharia Progressiva de Ontologias, detalhados a seguir.

Figura 12 – Visão Geral do *Framework* Proposto

4.4.1 Recursos do Domínio

Um Sistema Existente (1a) tem o conhecimento a seu respeito representado em dois grandes conjuntos de recursos: os Recursos de Entrada (1b) e os Recursos de Saída (1c). Os primeiros (1b) são aqueles utilizados na sua concepção como padrões e especificação do domínio e que, geralmente, estão expressos em textos

escritos em linguagem natural ou diagramas UML. Exemplos são o *Person Identification Service* (PIDS), no domínio da Saúde e o IMS LD, na Educação. Os organismos responsáveis por estes padrões disponibilizam os recursos em metadados padronizados e, muitas vezes, fazem parte do Repositório da ferramenta CASE (1d).

Os Recursos de Saída (1c) são produzidos durante o ciclo de vida de desenvolvimento do sistema, como os diversos diagramas UML, o modelo E/R, o Dicionário de Dados do SGBDR, os programas fontes e os textos e tutoriais disponibilizados para o usuário final.

Se, no desenvolvimento do Sistema Existente, foi utilizada alguma ferramenta CASE estes recursos de saída estão representados no Repositório da ferramenta (1d). Caso contrário a Engenharia Reversa (1e) deverá ser realizada para obtê-los.

Este módulo disponibiliza os modelos representados num Repositório (1d) e um conjunto de textos significativos sobre o Sistema Existente, que são fornecidos ao bloco Aprendizado de Ontologias.

4.4.2 Aprendizado de Ontologias

O bloco Aprendizado de Ontologias extrai e transforma, do conjunto de recursos destacados do domínio, representações iniciais da ontologia – metadados OWL. Está dividido em dois módulos: *schematas* e textos, que serão discutidos a seguir.

4.4.2.1 Schematas

Neste módulo, as instâncias (M0) dos Modelos de Dados (M1) – bloco (2a) - e dos diagramas UML (M1) – bloco (2b) -, representados no Repositório da ferramenta CASE (1d), são extraídas na forma de Metadado XMI (2a1 e 2b1), conforme representado na Figura 12.

Para obtenção de uma representação no metamodelo OWL (2e) é necessário realizar a Transformação entre Metamodelos (2c) utilizando a correspondência entre metamodelos apresentada nos Quadros 16 e 17. As associações entre os elementos ficam representadas no *framework* como conjunto de regras R1 (2a2) e R2 (2b2).

O conjunto de regras R1 (2a2) será aplicado nos metadados XMI (2a1) dos Modelos de Dados (E/R ou Relacional), de acordo com as primitivas do Modelo de Dados e considerando o modelo OWL conforme Quadro 16.

O conjunto de regras R2 (2b2) será aplicado nos Metadados XMI (2b1) dos diagramas UML, mais especificamente o diagrama de classes, de acordo com as primitivas da UML e considerando o modelo OWL conforme Quadro 17.

A Transformação entre Metamodelos no *framework* possuirá tantos subprocessos quantos modelos estiverem disponíveis no metamodelo das ferramentas utilizadas.

No final deste módulo obtêm-se duas ou mais representações iniciais da ontologia, denominadas Metadado OWL (2e).

4.4.2.2 A partir de Textos

Neste módulo, podem ser inseridas diversas ferramentas e diversos algoritmos para aprendizado de termos, conceitos, sinônimos, hierarquias, axiomas e/ou instâncias, a partir dos textos selecionados no primeiro bloco.

A Extração dos Elementos da ontologia (3a) a partir dos textos representa os resultados (parciais e finais) no formato considerado pelo metamodelo da ferramenta utilizada, Metadado (3b). Este processo pode utilizar uma ou mais das ferramentas existentes no AO a partir de textos (Quadro 8).

A Transformação entre Metamodelos (3c) aplica um conjunto de regras R3 (3b1) no metadado (3b) para obter representações no Metamodelo OWL (3d), segundo o Quadro 18. As regras R3 geralmente são definidas e construídas internamente nas ferramentas para AO.

Esse processo de Transformação entre Metamodelos no *framework*, tal como para extração de *schematas*, possuirá tantos subprocessos quantos metamodelos ou conjunto de primitivas diferentes existirem. No final deste módulo, obtêm-se uma representação inicial da ontologia denominada Metadado OWL (3d).

4.4.3 Ambiente de Ontologias

O Ambiente de Ontologias reúne o maior número de funcionalidades necessárias para o Tratamento de Ontologias (4a) e a construção final da ontologia, conforme o

Quadro 6, com destaque especial no *framework* para edição, população e importação.

Nesse bloco os Metadados OWL - (2e) e (3d) - são importados, validados, visualizados e integrados no módulo Tratamento de Ontologias (4a) de tal forma a obter uma representação inicial consolidada da ontologia (4b) disponível na Base Ontológica (5b). O Ambiente de Ontologias contempla o metamodelo ODM para OWL de forma a permitir a composição de várias ferramentas para o tratamento dos modelos da ontologia.

4.4.4 Engenharia Progressiva de Ontologias

O bloco Engenharia Progressiva de Ontologias recebe as representações iniciais no ambiente de ontologia e conduz o Método para Desenvolvimento (5a) de acordo com a ontologia a ser construída.

4.5 Requisitos para Participação

O Sistema Existente e o conjunto de ferramentas candidatas a participar do *framework* - CASE, Aprendizado de Ontologias a partir de textos, Transformação de Metamodelos e Ambiente de Ontologias – devem atender um conjunto de requisitos focados na representação do conhecimento existente do domínio, na interoperabilidade dos modelos entre as ferramentas e no aprendizado dos elementos da ontologia.

A interoperabilidade dos modelos é necessária para troca e transformação de modelos entre as ferramentas participantes do *framework* proposto. Para isso os requisitos precisam representar os metamodelos dos modelos considerados e realizar as transformações necessárias entre eles, ou ter a funcionalidade de extração e importação dos modelos no formato XMI. O ODM, metamodelo padrão para representação da ontologia, é fundamental para as saídas do bloco de Aprendizado de Ontologias e para todo o tratamento de ontologias necessário nos outros dois blocos: Ambiente de Ontologias e Engenharia Progressiva de Ontologias.

O enfoque de aprendizado de ontologias considera os requisitos necessários para aprender os elementos da ontologia do domínio em questão e transportá-los entre os diferentes metamodelos das ferramentas.

4.5.1 Sistemas Existentes

Os Sistemas Existentes são parte integrante do *framework* proposto, cujo objetivo é recuperar representações do conhecimento já existentes para compreender o domínio e obter indivíduos para a ontologia.

Os requisitos funcionais necessários para selecionar os Sistemas Existentes para o *framework* proposto estão destacados no Quadro 19.

	Sistemas Existentes		
	Requisitos referentes ao Recursos de Entrada e de Saída	Essencial	Desejável
E1	Possuir modelo de análise em UML num Repositório		x
E2	Possuir modelo E/R num Repositório		x
E3	Possuir dados implementados num SGBDR	x	
E4	Existir textos significativos	x	
E5	Seguir metadados padronizados do Domínio		x
E6	Existir metadados padronizados do Domínio expressos em UML		x
	Outras características		
E7	Ser representativo no Domínio	x	

Quadro 19 – Requisitos p/ Sistemas Existentes

Descrição dos requisitos:

- [E1] – O Sistema Existente poderá ter seus modelos de análise, representados em diagramas UML, num repositório de alguma ferramenta CASE.
- [E2] – O Sistema Existente poderá ter seus modelos de dados, representados pelo Modelo E/R e/ou Relacional, num repositório de alguma ferramenta CASE.
- [E3] – O Sistema Existente deverá ter seus dados implementados num SGBDR. Este requisito aliado aos requisitos da ferramenta CASE, C10 ou C11, permitirá realizar a engenharia reversa do Modelo Relacional.
- [E4] – O Sistema Existente deverá ter textos significativos disponíveis para realização do bloco de Aprendizado de Ontologias a partir de textos.
- [E5] – O Sistema Existente poderá ter seguido metadados ou recomendações padronizadas

- [E6] sobre o domínio que representa. Estes requisitos permitem a participação de textos ou *schematas* definidos por organismos responsáveis pela padronização do domínio e utilizados na concepção do sistema existente.
- [E7] – O Sistema Existente deverá ser representativo no domínio.

4.5.2 Ferramenta CASE

A ferramenta CASE tem como objetivos obter as representações existentes na forma de *schematas* - modelo de dados e diagramas UML – armazenando-as num repositório padrão e realizar as transformações dos modelos existentes para o ODM/OWL.

Os requisitos funcionais para seleccionar uma ferramenta CASE para o *framework* proposto estão destacados no Quadro 20.

	Ferramenta CASE		
	Requisitos referentes ao padrão OMG/MDA/MOF	Essencial	Desejável
C1	Contemplar modelos E/R com seus elementos e primitivas	x	
C2	Contemplar diagramas UML com seus elementos e primitivas	x	
C3	Possuir metamodelo dos modelos considerados no padrão MOF/M2	x	
C4	Importar modelos no padrão MOF		x
C5	Exportar modelos no padrão MOF/XMI	x	
C6	Transformar Modelo E/R para o Modelo Relacional		x
C7	Transformar Modelo Relacional para o Modelo E/R	x	
C8	Possuir Profile UML para ODM		x
C9	Transformar diagramas UML para OUP		x
	Requisitos referentes à Engenharia Reversa		
C10	Realizar Engenharia Reversa a partir de conexão com SGBDR		x
C11	Realizar Engenharia Reversa a partir de <i>scripts</i> SQL	x	
	Outros Requisitos		
C12	Possuir repositório dos modelos	x	

Quadro 20 – Requisitos p/ Ferramenta CASE

Descrição dos requisitos:

- [C1] – A ferramenta CASE deverá garantir a expressividade do modelo de dados, ou seja, contemplar todos os elementos e primitivas do Modelo E/R – entidades, atributos, relacionamentos, cardinalidade e instâncias - e do Modelo Relacional – tabelas, colunas, chaves estrangeiras e linhas conforme apresentado nas duas primeiras colunas do

Quadro 16.

- [C2] – A ferramenta CASE deverá garantir a expressividade dos diagramas UML principalmente para o diagrama de classes – classes, atributos, associações e outros elementos conforme apresentado na primeira coluna do Quadro 17.
- [C3] – A ferramenta CASE deverá considerar metamodelos padrão conforme as especificações do meta-metamodelo MOF para garantir a interoperabilidade entre os modelos das ferramentas participantes do *framework* proposto.
- [C4] – A ferramenta CASE poderá importar modelos construídos com outras ferramentas CASE com metamodelos padrão MOF através do tratamento dos modelos serializados em arquivos no formato XML. Este requisito permite, por exemplo, importar os mesmos modelos como Modelos E/R ou modelos diferentes como Modelos E/R para diagramas de classe da UML.
- [C5] – A ferramenta CASE deverá exportar seus modelos serializados em arquivos no formato XMI de acordo com as especificações padrão do meta-metamodelo - MOF.
- [C6] – A ferramenta CASE poderá realizar a transformação do Modelo E/R para o Modelo Relacional baseado nas especificações do metamodelo padrão MOF. Uma ferramenta CASE possui muitos metamodelos e nem sempre implementa todas transformações possíveis entre os metamodelos. Este requisito destaca a transformação do Modelo E/R para o Relacional.
- [C7] – A ferramenta CASE deverá realizar a transformação do Modelo Relacional para o Modelo E/R baseado nas especificações do metamodelo padrão MOF. Idem ao anterior.
- [C8] – A ferramenta CASE poderá possuir o metamodelo padrão para ontologia ODM. Este requisito será útil para refinar o modelo das representações de ontologias obtidas com o OUP (Profile UML para Ontologia) na Engenharia Progressiva.
- [C9] – A ferramenta CASE poderá transformar diagramas UML existentes em diagramas UML para ontologias, ou seja, aplicar as especificações do ODM nos diagramas existentes permitindo assim modelar a ontologia no seu ambiente ou exportar para outro ambiente já no formato padrão.
- [C10] – A ferramenta CASE poderá recuperar o Modelo Relacional representado fisicamente no dicionário do SGBDR do sistema existente. Este requisito considera o metamodelo do Modelo Relacional representado no dicionário de dados do SGBDR e efetua sua transformação para o metamodelo do Modelo Relacional representado na ferramenta.
- [C11] – A ferramenta CASE deverá recuperar o Modelo Relacional representado em *scripts* de criação da base de dados. Este requisito considera o metamodelo do Modelo Relacional representado em comandos SQL ANSI e efetua sua transformação para o metamodelo do Modelo Relacional representado na ferramenta.
- [C12] – A ferramenta CASE deverá armazenar as instâncias dos metamodelos num repositório, seguindo o padrão MOF, permitindo assim a extração dos modelos em arquivos no formato XMI bem como sua transformação para outros modelos.

4.5.3 Ferramentas para AO a partir de Textos

As ferramentas para AO a partir de Textos tem como objetivo aprender o maior número de elementos da ontologia, desde simples conceitos até relações não-taxonômicas e funções.

Os requisitos funcionais para selecionar uma ferramenta de Aprendizado de Ontologias a partir de Textos para o *framework* proposto estão destacados no Quadro 21.

	Aprendizado de Ontologias a partir de Textos		
	Requisitos referentes ao elemento aprendido	Essencial	Desejável
T1	Aprender conceitos	x	
T2	Aprender relações taxonômicas	x	
T3	Aprender relações não-taxonômicas		x
T4	Aprender instâncias de classes	x	
T5	Aprender instâncias de relações	x	
T6	Aprender funções e axiomas		x
	Requisitos referentes ao resultado		
T7	Expressar o resultado no formato OWL	x	
	Requisitos referentes ao tipo do ponto de início		
T8	Considerar textos específicos do domínio	x	
T9	Considerar textos em linguagem natural	x	
T10	Considerar documentos Web	x	
T11	Considerar documentos semi-estruturados (XML, DTD)		x
	Requisitos referentes ao padrão OMG/MDA/MOF		
T12	Disponibilizar metamodelo dos modelos considerados	x	
T13	Exportar modelos nos padrão ODM/MOF/M1 para OWL		x
T14	Exportar instâncias do modelo no padrão OWL	x	
	Outros Requisitos		
T15	Visualizar os resultados intermediários dos algoritmos		x
T16	Executar os algoritmos incrementalmente		x

Quadro 21 – Requisitos p/ ferramentas AO a partir de textos

Os requisitos T1 ao T6 foram colocados separados devido à existência de ferramentas específicas para cada elemento a ser aprendido, conforme constatado no Quadro 8.

Descrição dos requisitos:

- [T1] – A ferramenta para AO a partir de textos deverá extrair os conceitos relevantes do domínio presentes nos recursos textuais selecionados do sistema existente.
- [T2] – A ferramenta para AO a partir de textos deverá extrair as relações taxonômicas entre os conceitos do domínio presentes nos recursos textuais selecionados do sistema existente.
- [T3] – A ferramenta para AO a partir de textos poderá extrair as relações não-taxonômicas entre os conceitos do domínio presentes nos recursos textuais selecionados do sistema existente.
- [T4] – A ferramenta para AO a partir de textos deverá extrair instâncias de classes presentes nos recursos textuais selecionados do sistema existente.
- [T5] – A ferramenta para AO a partir de textos deverá extrair instâncias de relações presentes nos recursos textuais selecionados do sistema existente.
- [T6] – A ferramenta para AO a partir de textos poderá extrair funções e axiomas presentes nos recursos textuais selecionados do sistema existente.
- [T7] – A ferramenta para AO a partir de textos deverá disponibilizar o modelo de ontologia obtido no modelo OWL. Este requisito contempla a definição dos elementos da ontologia conforme Quadro 18.
- [T8] – A ferramenta para AO a partir de textos deverá considerar recursos no formato texto,
- [T9] linguagem natural e documentos Web sobre o domínio e apresentar resultados
- [T10] intermediários com grau de certeza ou relevância dos elementos da ontologia obtidos.
- [T11] – A ferramenta para AO a partir de textos poderá considerar documentos semi-estruturados sobre o domínio como XML e DTD.
- [T12] – A ferramenta para AO a partir de textos deverá disponibilizar seu metamodelo para representação de seus modelos. Este requisito é essencial porque a maioria das ferramentas não considera um metamodelo padrão existindo assim a necessidade de construir as regras para as ferramentas de transformação entre metamodelos. A existência deste requisito facilitará muito a interoperabilidade com as demais ferramentas do *framework* proposto.
- [T13] – A ferramenta para AO a partir de textos poderá exportar os modelos obtidos no ODM para OWL, ou seja, já possui a funcionalidade para transformação entre metamodelos.
- [T14] – A ferramenta para AO a partir de textos deverá exportar as instâncias da ontologia no formato OWL.
- [T15] – A ferramenta para AO a partir de textos poderá permitir a visualização dos resultados intermediários dos seus diversos algoritmos. Isto permitirá a aquisição progressiva de conhecimento sobre o domínio por parte do engenheiro de ontologias;
- [T16] – A ferramenta para AO a partir de textos poderá permitir executar incrementalmente seus diversos algoritmos implementados.

4.5.4 Ferramentas para Transformação entre Metamodelos

As ferramentas para Transformação entre Metamodelos têm como objetivo garantir o mapeamento do maior número de elementos representativos de um modelo para outro. Estas ferramentas participam no bloco de AO tanto para *schematas* como para Textos e são necessárias quando as ferramentas possuírem metamodelos distintos e/ou não realizarem internamente as transformações necessárias.

Os requisitos funcionais para selecionar estas ferramentas para o *framework* proposto estão destacados no Quadro 22.

Transformação entre Metamodelos (<i>schematas</i>, Textos)			
Requisitos referentes aos mapeamentos		Essencial	Desejável
M1	Possuir regras de mapeamento entre elementos Modelo E/R e o Modelo OWL de acordo com o Quadro 16	x	
M2	Possuir regras para mapeamento entre elementos dos Diagramas UML e Modelo OWL de acordo com o Quadro 17	x	
M3	Permitir criar regras para mapeamento entre as primitivas do modelo das ferramentas AO para modelo OWL de acordo com o Quadro 18	x	
Requisitos referentes ao padrão OMG/MDA/MOF			
M4	Implementar o ODM para o modelo OWL		x
M5	Exportar os resultados para o modelo destino no formato XMI		x
M6	Exportar o resultado no modelo OWL	x	
Outros			
M7	Interface gráfica		x
M8	Repositório de modelos no formato padrão XMI (entrada e saída)		x
M9	Considerar versões dos padrões (MOF/XMI/OWL)		x

Quadro 22 – Requisitos p/ ferramentas de transformação entre metamodelos

Descrição dos requisitos:

- [M1] – A ferramenta para Transformação de Metamodelos deverá possuir regras de mapeamento definidas entre os elementos do Modelo E/R e os elementos do modelo OWL de acordo com as especificações para metamodelos presentes no ODM e representadas no Quadro 16.
- [M2] – A ferramenta para Transformação de Metamodelos deverá possuir regras de mapeamento definidas entre os elementos dos diagramas UML e os elementos do modelo OWL de acordo com as especificações para metamodelos presentes no ODM e

representadas no Quadro 17.

- [M3] – A ferramenta para Transformação de Metamodelos deverá permitir a elaboração de regras para mapeamento entre as primitivas do modelo das ferramentas AO (seu metamodelo) para modelo OWL de acordo com o Quadro 18.
- [M4] – A ferramenta para Transformação de Metamodelos poderá implementar o ODM para OWL. Este requisito considera a existência dos metamodelos bem como das regras de transformação internamente na ferramenta.
- [M5] – A ferramenta para Transformação de Metamodelos poderá exportar o resultado em arquivos no formato XMI.
- [M6] – A ferramenta para Transformação de Metamodelos deverá exportar o resultado no modelo OWL.
- [M7] – A ferramenta para Transformação de Metamodelos poderá ter uma interface gráfica para informar os modelos de entrada e saída bem como as regras de mapeamentos a serem utilizadas.
- [M8] – A ferramenta para Transformação de Metamodelos poderá ter um repositório dos modelos, no formato padrão XMI, tanto para os modelos de entrada como para os modelos de saída. Este requisito visa manter as versões dos modelos considerados e das regras aplicadas – especificação considerada.
- [M9] – A ferramenta para Transformação de Metamodelos poderá considerar as versões das especificações padrões utilizadas (MOF/XMI/OWL) nas transformações realizadas. Este requisito visa manter as versões dos modelos considerados e das regras aplicadas – especificação considerada.

4.5.5 Ferramentas para o Ambiente de Ontologias

As ferramentas para o Ambiente de Ontologias têm como objetivo disponibilizar as funcionalidades necessárias para a edição, validação, integração e visualização das representações de ontologias construídas, entre outras (detalhes nos itens 2.5.3, 2.6.3 ou Quadros 6 e 8).

Os requisitos funcionais necessários para selecionar as ferramentas participantes deste Ambiente de Ontologias para o *framework* proposto estão destacados no Quadro 23.

	Ambiente de Ontologias		
	Requisitos referentes ao padrão OMG/MDA/MOF	Essencial	Desejável
O1	Possuir metamodelo no padrão MOF/M2		x
O2	Exportar modelos no padrão ODM/MOF/M1		x
O3	Importar modelos E/R de acordo com o ODM para OWL		x
O4	Importar diagramas UML de acordo com o ODM para OWL		x
O5	Importar modelos OWL	x	
	Outras funcionalidades		
O6	Editar as ontologias	x	
O7	Integrar ontologias	x	
O8	Visualizar graficamente a representação da ontologia		x
O9	Validar a representação formal e consistência da ontologia	x	
O10	Apresentar ontologia graficamente		x
O11	Importar ontologia no formato OWL	x	
O12	Ter repositório para os modelos		x

Quadro 23 – Requisitos p/ Ambiente de Ontologias

Descrição dos requisitos:

- [O1] – A ferramenta para o Ambiente de Ontologias poderá possuir ODM para OWL para recuperar modelos de ontologia construídos em outras ferramentas.
- [O2] – A ferramenta para o Ambiente de Ontologias poderá exportar seus modelos em arquivos no formato padrão XML - considerando a presença do requisito O1.
- [O3] – A ferramenta para o Ambiente de Ontologias poderá importar diretamente Modelos E/R de acordo com o ODM para OWL. Este requisito considera a existência do requisito O1 e a presença do metamodelo do Modelo E/R na ferramenta.
- [O4] – A ferramenta para o Ambiente de Ontologias poderá importar diagramas UML de acordo com o ODM para OWL. Este requisito considera a existência do requisito O1 e a presença dos metamodelos dos diagramas UML na ferramenta.
- [O5] – A ferramenta para o Ambiente de Ontologias deverá importar modelos OWL. Este requisito garante a existência do metamodelo para o modelo OWL entre os metamodelos da ferramenta.
- [O6] – A ferramenta para o Ambiente de Ontologias deverá editar o modelo da ontologia importado. Este requisito permite a manutenção do modelo obtido como incluir novos elementos, alterá-los ou excluí-los.
- [O7] – A ferramenta para o Ambiente de Ontologias deverá integrar os modelos OWL obtidos. Este requisito é fundamental para a integração das diversas representações obtidas do bloco de Aprendizado de Ontologias. Existem várias ferramentas com diferentes

abordagens para realizarem esta funcionalidade.

- [O8] – A ferramenta para o Ambiente de Ontologias poderá visualizar a representação da ontologia obtida. Este requisito permite ao usuário ter uma visão gráfica tanto da ontologia.
- [O9] – A ferramenta para o Ambiente de Ontologias deverá validar a representação da ontologia obtida. Este requisito permite verificar as estruturas sintáticas obtidas e sua completeza em relação ao modelo OWL.
- [O10] – A ferramenta para o Ambiente de Ontologias poderá apresentar as instâncias da ontologia graficamente.
- [O11] – A ferramenta para o Ambiente de Ontologias deverá importar instâncias da ontologia no formato OWL.
- [O12] – A ferramenta para o Ambiente de Ontologias poderá ter um repositório para armazenar seus modelos e versões.

4.6 Atividades do *framework* proposto

As atividades a serem realizadas pelo engenheiro de ontologias para utilizar o *framework* proposto, conforme Figura 12, estão graficamente representadas no diagrama de atividades apresentado na Figura 13.

Figura 13 – Diagrama de Atividades do *framework* proposto

4.6.1 Selecionar Recursos do Domínio

O engenheiro de ontologias irá separar e classificar os Recursos de Entrada e de Saída presentes no Sistema Existente. Para isso, o engenheiro de ontologias realiza o levantamento dos recursos existentes e os separa de acordo com o tipo de dado considerado no AO: ontologias, *schemata*, instâncias, documentos Web ou dados semi-estruturados. Assim, agrupará *schematas* - diagramas UML e modelo de dados – e textos - documentos Web e dados semi-estruturados presentes na documentação do sistema e em tutoriais.

Para os *schematas* existem duas possibilidades:

- já possuir representação no Repositório da ferramenta CASE,

- existir representação nas ferramentas que os implementaram, como o ambiente de desenvolvimento e/ou SGBDR.

Na primeira possibilidade, o engenheiro de ontologias precisará verificar se a ferramenta CASE utilizada já considera os padrões MDA e MOF e se possui a funcionalidade de extração dos modelos no formato XMI. No caso de não considerar os padrões, então os modelos precisarão ser transportados para uma ferramenta que considere esses padrões.

Na segunda possibilidade, quando os *schematas* não estão representados num Repositório, o engenheiro de ontologias terá de realizar o processo de Engenharia Reversa (1e) para obtê-los. Este processo prevê a utilização de uma ferramenta CASE baseada no padrão MDA/MOF, de tal forma que, uma vez realizada a reengenharia, as instâncias do modelo possam ser obtidas no formato XMI. A Engenharia Reversa é aplicada no SGBDR para obtenção do modelo de dados (E/R e Relacional) e no ambiente de desenvolvimento para obtenção dos diagramas UML. O engenheiro de ontologias agrupa os textos presentes nos Recursos de Entrada e nos Recursos de Saída como Tutoriais, Textos e outros. Estes recursos (agrupados) servirão como entrada no processo de Extração dos Elementos da Ontologia (3a), dentro do bloco de Aprendizado de Ontologias.

4.6.2 Selecionar Ferramentas

O engenheiro de ontologias deverá verificar os requisitos para as ferramentas, escolher as ferramentas adequadas para o tratamento dos recursos agrupados e compor o *framework* para utilização. As ferramentas para participarem do *framework* devem ser selecionadas de acordo com o atendimento aos requisitos enumerados no *framework* proposto, principalmente, a representatividade do seu metamodelo, a utilização dos padrões considerados somado à disponibilidade de uso.

O engenheiro de ontologias executará esta atividade de acordo com o tipo dos recursos agrupados e as ferramentas disponíveis no ambiente do Sistema Existente e no seu ambiente de trabalho. Em alguns casos serão necessárias ferramentas para transformação entre metamodelos, noutros ferramentas para aprendizado de ontologias a partir de textos e noutros todos os recursos do *framework*.

As representações iniciais visam facilitar o entendimento sobre o domínio pelo engenheiro de ontologias e permitir discussões mais elaboradas com o especialista de domínio. Esta construção incremental do entendimento pode levar o engenheiro de ontologias a adicionar novas ferramentas ao *framework* conforme o nível da ontologia almejada e os recursos disponíveis do domínio.

4.6.3 Realizar o Aprendizado de Ontologias

Com os recursos agrupados e as ferramentas selecionadas o engenheiro de ontologias realiza o Aprendizado de Ontologias conforme descrito no bloco Aprendizado de Ontologia da visão geral do *framework*.

Um aspecto importante desta atividade é sua simplicidade de execução. O engenheiro de ontologias não precisará conhecer os detalhes de implementação dos algoritmos das diversas ferramentas, somente sua instalação, utilização e obtenção do resultado final. O objetivo é aprender gradativamente sobre o domínio em questão conforme as camadas propostas por Cimiano (2006) na Figura 5.

4.6.4 Utilizar o Ambiente de Ontologias

O engenheiro de ontologias já deve ter um ambiente de ontologias previamente selecionado, ou incrementado, conforme os requisitos para participação do *framework*. Uma vez obtidas as representações iniciais da ontologia, estas representações precisam ser importadas no ambiente, validadas, editadas, visualizadas e manipuladas.

Na importação erros de sintaxe podem ser detectados por inconsistências no processo de extração ou transformação entre metamodelos. No primeiro caso a representação já apresenta erros no ambiente de aprendizado de ontologias e no segundo caso a ferramenta de transformação pode não ter executado corretamente ou considerado as primitivas corretamente nas transformações.

W3C detalhou as regras sintáticas, conformidade semântica e outros fatores para validar os artefatos produzidos em OWL⁶⁷ e classificá-los segundo as situações encontradas.

Na edição e visualização o engenheiro de ontologias poderá identificar facilmente os elementos da ontologia como conceitos, propriedades e instâncias.

4.6.5 Selecionar Método para Engenharia Progressiva

A seleção do Método para Engenharia Progressiva está associada ao tipo de ontologia a ser construída, ao desenvolvimento ser centralizado ou descentralizado, e às atividades de Aprendizado de Ontologias estarem inseridas nos estágios do método, principalmente, no estágio de conceitualização.

O tipo de ontologia - de domínio, de tarefa, de aplicação, conforme exposto no Quadro 1 - determina o nível de consenso necessário para a ontologia. A Figura 14 mostra isto em termos de reusabilidade e interoperabilidade considerando alguns tipos de ontologia – Aplicação, Domínio e Geral – numa pirâmide, onde na base encontra-se a Representação da Ontologia. Assim uma ontologia de Aplicação tem maior usabilidade e menor reusabilidade, enquanto uma ontologia Geral, como as destinadas aos serviços semânticos, tem maior reusabilidade e menor usabilidade. Uma ontologia Geral é mais abrangente e precisa garantir a interoperabilidade de suas representações entre as diversas ferramentas presentes, tanto na sua utilização como na sua modelagem, exigindo assim um nível maior de consenso entre os especialistas.

Figura 14 – Ontologias, Usabilidade e Reusabilidade, adaptada (Gómez-Pérez, 2007)

⁶⁷ <http://www.mygrid.org.uk/OWL/Validator>

Nos métodos centralizados, os engenheiros de ontologia compartilham o mesmo ambiente e as mesmas ferramentas, tornando a introdução das atividades de Aprendizado de Ontologia e transformações de metamodelos mais localizadas. Já os métodos descentralizados, estão mais preparados para integrar diversas representações de ontologias e validá-las, pois facilitam a colaboração entre pessoas em diferentes locais, combinam a utilização de um mesmo padrão e tratam várias construções intermediárias durante as discussões relevantes para a conceitualização.

Atividades de AO devem fazer parte das atividades previstas nas fases de conceitualização e formalização da ontologia. É nessas fases que os principais conceitos do domínio e seus relacionamentos estão sendo definidos e axiomas estão sendo formulados para restringir a possibilidade de interpretação do significado do conceito. Neste momento, pode-se combinar as abordagens de construção manual e semi-automática de ontologias. Os processos de AO – importar, extrair, aparar e refinar - devem ser inseridos nessas fases.

4.7 Discussão

O *framework* proposto evidencia as vantagens no uso de metamodelos padronizados no AO a partir de *schematas* – MDA, UML e MOF - e propõe o mesmo para AO a partir de textos – MDA, ODM e MOF – e para o Ambiente de Ontologias. Assim, os resultados das pesquisas em PLN e Aprendizagem de Máquinas, que estão em constante evolução, podem ser aproveitados por um número maior de ferramentas do ambiente de ontologias e colaborarem para a construção de representações de ontologias para a WS. O ambiente de ontologias é um conjunto de ferramentas heterogêneas em suas funcionalidades – edição, validação, visualização, junção, integração entre outras – e não tem um metamodelo padrão. Ao sugerir a adoção dos padrões MDA, ODM e OWL para este ambiente, o *framework* proposto resolve o problema de interoperabilidade entre elas e também com as ferramentas de AO. Os modelos utilizados não são equivalentes mas complementares.

A estrutura do *framework* proposto pode ser considerada uma visão nova da arquitetura proposta por Maedche (2002) no projeto KAON para AO, mostrados na

Figura 6, e considerou, principalmente, seus módulos para Processamento de Recursos, Conjunto de Resultados e Coordenação do AO. Comparando as atividades selecionadas com a metodologia no projeto KAON (Figura 6), as atividades presentes no bloco “Realizar AO” da Figura 13 correspondem ao Processamento de Recursos (3) e as atividades no bloco “Utilizar o Ambiente de Ontologias” ao Gerenciamento Gráfico (1). O diferencial neste *framework* proposto são os recursos de entrada - limitados aos recursos de sistemas existentes de um dado domínio -, a possibilidade de participação de diversas ferramentas para AO - não limitadas às do projeto KAON – e a aderência aos metamodelos padrão para *schematas* e textos.

Na Engenharia de Ontologias (Figura 3), estas atividades estariam distribuídas no bloco de Desenvolvimento estando as atividades do bloco “Selecionar Recursos do Sistema Existente” na Análise de Domínio, as do bloco “Realizar AO” na Conceitualização e as do bloco “Utilizar o Ambiente de Ontologias” distribuídas na Implementação, Aquisição do Conhecimento e Avaliação. Estas duas últimas atividades se ligam ao Suporte.

Diferentes propostas para AO podem ser encontradas na literatura e apresentam diferenças, principalmente na abrangência contemplada pelo *framework* proposto.

GAŠEVIĆ et al. (2004) modelaram as ontologias em OUP com a ferramenta Poseidon, implementaram a transformação dos modelos obtidos para OWL utilizando XSLT com uma folha de estilo denominada UMLtoOWL e trataram ontologia no formato OWL com a ferramenta Protégé. Esta implementação, apesar de compartilhar alguns recursos e ferramentas, é diferente do proposto neste trabalho uma vez que os recursos existentes de entrada, mais especificamente os *schematas*, não foram construídos com OUP.

Asium (FAURE; POIBEAU, 2000) também é um ambiente cooperativo, gráfico e amigável para aquisição de ontologias, porém contempla somente o aprendizado de relações taxonômicas mediante a aplicação de técnicas do aprendizado de máquinas.

OntoLT (BUIBELAAR; OLEJNIK; SINTEK, 2004) é uma ferramenta para construção de ontologias que realiza o aprendizado a partir de textos anotados na ferramenta Schug em XML. A seleção dos termos é realizada através de consultas definidas pelo usuário e funciona como um plug-in da ferramenta Protégé.

OntoLearn (NAVIGLI; VELARDI, 2006) é uma arquitetura para aprendizado automático de ontologias a partir de texto. O primeiro passo consiste de extrair a terminologia a partir de textos do domínio utilizando PLN e técnicas estatísticas e depois é realizada a interpretação dos termos com WordNet para extração de relações taxonômicas entre os conceitos.

Estas propostas resultam da diversidade de técnicas e métodos do AO e de um espectro cada vez mais amplo para o uso de ontologias.

O *framework* proposto procura compor diferentes ferramentas para AO integradas a diferentes ferramentas necessárias no ambiente de ontologias com base no metamodelo OWL e considerando a importância das atividades da Engenharia Progressiva de Ontologias.

5 PROVA DE CONCEITO NO DOMÍNIO DA EDUCAÇÃO

Diante das necessidades de ontologias para WSE e dos inúmeros sistemas educacionais existentes, o domínio da Educação torna-se um bom exemplo para realizar a prova de conceito do *framework* proposto. O objetivo desta prova de conceito é validar as idéias propostas, verificar se as atividades propostas, os padrões utilizados e os requisitos para as ferramentas do *framework* aplicam-se num ambiente real, de acordo com o sugerido conceitualmente e analisar os resultados obtidos. Este capítulo divide-se na aplicação do *framework* proposto e na análise e discussão dos resultados obtidos.

5.1 Aplicação do *framework* proposto

Para obter entendimento inicial do domínio foram considerados sistemas educacionais conforme o modelo de referência arquitetural apresentado na Figura 9, sendo dois LMS e um ambiente de autoria.

Os LMS utilizados por grandes universidades brasileiras selecionados foram: o Teleduc utilizado pela UNICAMP e o COL utilizado pela USP. O COL encontra-se, no momento, em fase de substituição pelo Tidia-Ae.

Teleduc é um software livre para um ambiente de criação, participação e administração de cursos na Web. Dentre as características desse ambiente pode-se citar o controle de participantes, de materiais e de troca de mensagens, as ferramentas para administrar cursos, o material de apoio disponível, o controle da autoria, a comunicação através de correio eletrônico entre os participantes, a formação de grupos de discussão, entre outros.

COL é uma ferramenta gerenciadora de cursos via Web que apresenta funcionalidades envolvendo aluno, turmas, disciplinas, matrículas, listas de discussão, e conteúdos da disciplina com fóruns, atividades, testes, avaliações e troca de mensagens.

Os dois LMS foram selecionados na busca pela identificação de mesmos conceitos do domínio presentes em modelos distintos desenhados para funcionalidades similares.

O Reload é um ambiente de autoria, tanto de Objetos de Aprendizagem como Projetos de Aprendizagem, com edição de metadados educacionais como IMS *Metadata*, IEEE LOM *Metadata*, SCORM, IMS *Content* e IMS LD.

O Reload foi utilizado para autoria do Curso de Computação Gráfica do INTERLAB segundo as práticas do IMS LD e cujos detalhes encontram-se no Anexo E.

Os requisitos do *framework* proposto para os sistemas existentes estão destacados no Quadro 24. O hífen indica falta de conhecimento sobre o atendimento ou não do requisito.

	Sistemas Existentes	COL	Teleduc	Reload
	Requisitos referentes aos Recursos de Entrada e de Saída			
E1	Possuir modelo de análise em UML num Repositório	-	-	Sim
E2	Possuir modelo E/R num Repositório	Sim	Sim	Não
E3	Possuir dados implementados num SGBDR	Sim	Sim	Não
E4	Existir textos significativos	Sim	Sim	Sim
E5	Seguir metadados padronizados do Domínio	-	-	Sim
E6	Existir metadados padronizados do Domínio expressos em UML	-	-	Sim
	Outras características			
E7	Ser representativo no Domínio	Sim	Sim	Sim

Quadro 24 – Sistemas Educacionais e requisitos do *framework* proposto

Uma vez disponibilizados os recursos dos sistemas educacionais, as atividades do *framework* proposto foram realizadas conforme detalhadas nos próximos itens, com referências às Figuras 12 e 13.

5.1.1 Selecionar Recursos do Domínio

Os *schematas* agrupados foram os *scripts* de criação do SGBDR dos sistemas COL e Teleduc e os diagramas de classes do modelo conceitual do IMS LD. Os gerenciadores do sistema COL disponibilizou o acesso ao SQL Server, e foi realizada a extração do *script* de criação do banco de dados com um utilitário do próprio SGBDR. O grupo de suporte do sistema Teleduc enviou os *scripts* de criação do banco de dados após solicitação formal. O IMS LD encontra-se representado e disponível no *DSpace* da *Open Universiteit Nederland*⁶⁸ - um repositório para

⁶⁸ <http://dspace.ou.nl/handle/1820/659>

resultados digitais de pesquisas desenvolvido pelo *Massachusetts Institute of Technology* (MIT) e Hewlett-Packard (HP) .

Os repositórios das ferramentas CASE utilizados na construção dos sistemas não foram solicitados, ou disponibilizados, porém os *schematas* obtidos têm formato padrão SQL ANSI para os *scripts* e padrão XML para o diagrama de classes. Isto permite a preparação do Repositório (1d) do *framework* proposto através da Engenharia Reversa (1e) na ferramenta CASE a ser selecionada.

Os recursos textuais agrupados foram os textos provenientes do modelo de informação do Modelo Conceitual do padrão IMS LD⁶⁹.

5.1.2 Selecionar Ferramentas

As ferramentas participantes do *framework* foram selecionadas de acordo com o atendimento aos requisitos enumerados no *framework* proposto. Elas estão descritas a seguir.

5.1.2.1 CASE

As ferramentas selecionadas foram MagicDraw⁷⁰ para os recursos de entrada do Teleduc e COL e Poseidon⁷¹ para o Reload. Ambas atendem a todos os requisitos essenciais enumerados no Quadro 25.

⁶⁹ http://www.imsglobal.org/learningdesign/ldv1p0/imsld_bestv1p0.html

⁷⁰ <http://www.magicdraw.com/>

⁷¹ <http://www.gentleware.com/index.php?id=ce>

	Ferramenta CASE	MagicDraw	Poseidon
	Requisitos referentes ao padrão OMG/MDA/MOF		
C1	Contemplar modelos E/R com seus elementos e primitivas	Sim	Sim
C2	Contemplar diagramas UML com seus elementos e primitivas	Sim	Sim
C3	Possuir metamodelo dos modelos considerados no padrão MOF/M2	Sim	Sim
C4	Importar modelos no padrão MOF	Sim	Sim
C5	Exportar modelos no padrão MOF/XMI	Sim	Sim
C6	Transformar Modelo E/R para o Modelo Relacional	Sim	Sim
C7	Transformar Modelo Relacional para o Modelo E/R	Sim	Sim
C8	Possuir Profile UML para ODM	Sim	Sim
C9	Transformar diagramas UML para OUP	Sim	Sim
	Requisitos referentes à Engenharia Reversa		
C10	Realizar Engenharia Reversa a partir de conexão com SGBDR	Sim	Sim
C11	Realizar Engenharia Reversa a partir de <i>scripts</i> SQL	Sim	Sim
	Outros Requisitos		
C12	Possuir repositório dos modelos	Sim	Sim

Quadro 25 – CASE e requisitos

No *site* OMG, encontra-se uma lista dos fornecedores de ferramentas⁷² no padrão MDA, isto é, que compartilham o meta-metamodelo, seus metamodelos e, conseqüentemente, seus modelos. Dentre eles destacam-se: IBM com *Rational Software Development Platform*, Telelogic com Telelogic TAU®, Gentleware com *Poseidon* e Sybase com *PowerDesigner*. Outras listas⁷³ indicam, além da descrição, a plataforma e o preço das ferramentas. A maioria das ferramentas relacionadas nestas listas atende aos requisitos do *framework* proposto.

5.1.2.2 Aprendizado de Ontologias a partir de Textos

Diante dos diversos sistemas de aprendizado, apresentados no Quadro 8, e dos requisitos essenciais para participação do *framework* proposto, listados no Quadro 21, foi escolhido o Text2Onto da Universidade de Karlsruhe. Além de atender aos requisitos essenciais, o Text2Onto possui uma variedade de algoritmos de

⁷² <http://uml-directory.omg.org/vendor/list.htm>

⁷³ http://www.objectsbydesign.com/tools/umltools_byCompany.html

aprendizado agrupados por elemento a ser aprendido, apresenta uma seqüência de aprendizado progressiva, conforme sugerido por Cimiano na Figura 5, e já é utilizado pela comunidade acadêmica desde 2000, quando era denominado TexToOnto.

Outra importante característica da ferramenta para o *framework* proposto é representar a estrutura ontológica aprendida na forma de meta-nível, chamada de primitivas de modelagem, ao invés de uma linguagem de representação concreta. O *Learned Ontology Model* (LOM) no Text2Onto denomina-se *Probabilistic Ontology Model* (POM) e garante a independência do formalismo bem como o poder de expressividade da linguagem destino para os elementos da ontologia aprendidos. As primitivas deste metamodelo estão presentes na *Modeling Primitive Library* (MPL) e são as seguintes: conceitos, hierarquia de conceitos, instanciação de conceitos, propriedades gerais e relações, relações mereológicas, restrições de domínio e limite e equivalência. Cada instância de uma das primitivas recupera um número médio de anotações, indicando qual a certeza do algoritmo de aprendizado executado sobre a relevância e certeza da instância correspondente dentro do conjunto de textos considerados. O objetivo destas “probabilidades” é facilitar ao usuário a consideração e seleção dos elementos aprendidos. Este metamodelo permite a transformação entre várias linguagens como RDFS, OWL e *F-Logic*. Text2Onto já foi integrado com outros ambientes de ontologias como o OntoEdit e o KAON2.

5.1.2.3 Ambiente de Ontologias

Para o ambiente de ontologias a ferramenta Protégé foi selecionada dentre as existentes no Quadro 6 e os requisitos para participação do *framework* proposto podem ser observados no Quadro 26.

	Ambiente de Ontologias	Protégé
	Requisitos referentes ao padrão OMG/MDA/MOF	
O1	Possuir metamodelo no padrão MOF/M2	Não
O2	Exportar modelos no padrão ODM/MOF/M1	Não
O3	Importar modelos E/R de acordo com o ODM para OWL	Não
O4	Importar diagramas UML de acordo com o ODM para OWL	Sim
O5	Importar modelos OWL	Sim
	Outras funcionalidades	
O6	Editar as ontologias	Sim
O7	Integrar ontologias	Não
O8	Visualizar graficamente a representação da ontologia	Sim
O9	Validar a representação formal e consistência da ontologia	Sim
O10	Apresentar ontologia graficamente	Sim
O11	Importar ontologia no formato OWL	Sim
O12	Ter repositório para os modelos	Sim

Quadro 26 – Protégé e os requisitos do *framework* proposto

Como o Protégé não possui seu metamodelo no padrão MOF/M2 torna-se necessário conhecer e ter acesso ao seu metamodelo para realização das importações e exportações dos modelos.

A arquitetura do Protégé está separada em parte de modelo e parte de visão. A parte de modelo é a representação interna dos mecanismos para ontologias - metamodelo utilizado pela ferramenta - e base de conhecimento. A parte de visão tem componentes para interface visual e manipulação do modelo.

A parte de modelo do Protégé é baseada num metamodelo simples e flexível (NOY; FERGERSON; MUSEN, 2000; PROTÉGÉ, 2006) com os elementos básicos sendo classes (*Class*), propriedades (*Slot*), características e restrições de propriedades (*Facet*), instâncias dos valores das propriedades (*Instance*), características padrão (*StandardFacet*), propriedades padrão (*StandardSlot*) e metaclasses (*StandardClass*), conforme mostra a Figura 15. Outras informações são obtidas da leitura do modelo como, as restrições das propriedades estão definidas na classe *StandardSlot*, e a seta de navegabilidade apontando da classe fonte *Class* para a classe alvo *StandardClass* indica que uma *Class* é uma subclasse de *StandardClass*. Esta estrutura do metamodelo e um conjunto de API escrito em *Java Metadata Interface* para acessá-lo, garantem a construção de extensões e adaptações para outras

Dois *plug-ins* foram utilizados neste trabalho: Jambalaya e XMI Backend. Jambalaya é um visualizador gráfico das ontologias. *XMI Backend* permite a interoperabilidade dos modelos construídos na ferramenta Protégé com outras ferramentas.

5.1.2.4 Transformação entre Metamodelos

A correspondência entre metamodelos é um componente crítico para o *framework* e as ferramentas para transformação devem programar as regras para manter o máximo de expressividade entre os modelos transformados. Existem várias linguagens e especificações disponíveis, umas oriundas de organizações para padrões como MOF QVT da OMG⁷⁴ e XSLT da W3C, e outras de organizações independentes como ATL (INRIA), VIATRA, o GReAT, Tefkat, KM3 e Kermeta. Estas iniciativas realizam as transformações entre metamodelos, principalmente, entre CIM₁ e PIM e PIM e PSM. O *framework* proposto deverá aproveitar ferramentas já construídas.

O Quadro 27 relaciona as ferramentas utilizadas nas transformações versus os requisitos do *framework* proposto. As ferramentas próprias para transformação – Xalan, ATL e CIMTool - estão preparadas para mapeamentos dos *schematas*. As ferramentas para manipulação de ontologia – Protégé – e para aprendizado de ontologias - Text2Onto – apresentam funcionalidades para transformação entre metamodelos.

⁷⁴ <http://www.omg.org/docs/ptc/07-07-07.pdf>

	Transformação entre Metamodelos (<i>schematas</i>, Textos)	Xalan	ATL	CIMTool	Protégé	Text2Onto
	Requisitos referentes aos mapeamentos					
M1	Possuir regras de mapeamento entre elementos Modelo E/R e o Modelo OWL de acordo com o Quadro 22	Sim	Sim	Sim	Não	Não
M2	Possuir regras para mapeamento entre elementos dos Diagramas UML e Modelo OWL de acordo com o Quadro 23	Sim	Sim	Sim	Sim	Não
M3	Permitir criar regras para mapeamento entre as primitivas do modelo das ferramentas AO para modelo OWL de acordo com o Quadro 24	Não	Não	Não	Não	Sim
	Requisitos referentes ao padrão OMG/MDA/MOF					
M4	Implementar o ODM para o modelo OWL	Sim	Sim	Sim	Não	Não
M5	Exportar os resultados para o modelo destino no formato XMI	Sim	Sim	Sim	Não	Não
M6	Exportar o resultado no modelo OWL	Sim	Sim	Sim	Sim	Sim
	Outros					
M7	Interface gráfica	Não	Não	Sim	Sim	Sim
M8	Repositório de modelos no formato padrão XMI (entrada e saída)	Não	Sim	Não	Não	Não
M9	Considerar versões dos padrões (MOF/XMI/OWL)	Não	Não	Não	Não	Não

Quadro 27 – Transformações e requisitos

O processador Xalan é escrito em XSLT e seu funcionamento está baseado na definição de regras de transformação expressas num documento chamado folha de estilo (*XSL Style Sheet*). Estas regras são aplicadas no arquivo de entrada (formato XML), produzindo outro arquivo transformado também no formato XML. Já existe uma folha de estilo para, junto com arquivos XMI representando modelos UML, ser utilizada na transformação para o metamodelo OWL, denominada UMLtoOWL.

Em ATL foram desenvolvidos dois cenários de transformação envolvendo MDA e OWL: um chamado UML2OWL, onde a entrada é um arquivo XMI, obtido a partir de algum modelo UML e a saída é uma representação da ontologia no metamodelo

OWL; o outro é um extrator XML para produzir documentos XML conforme a sintaxe OWL.

CIMTool é também uma ferramenta de código aberto destinada a apoiar as transformações necessárias entre os padrões CIM₂ (*Common Information Model*), usados na indústria de eletricidade. Esta ferramenta lê e agrupa modelos UML no formato XMI, verifica inconsistências e gera a ontologia equivalente em OWL. Na sua versão CIMTool20070509 permite as versões XMI 1.1 e 1.2, os diagramas UML 1.4 e OWL *Lite*.

Umt-qvt⁷⁵ e Modelware⁷⁶ são outras ferramentas para transformação e não foram consideradas neste trabalho pelo fato das ferramentas anteriores já terem apresentado resultados satisfatórios.

Protégé permite a transformação entre metamodelos, tanto na importação quanto na exportação de modelos. Na criação de novos projetos permite escolher o tipo de projeto (*Protégé Files*, *Protégé Database*, *Experimental XML File (.xml)*, *OWL / RDF Database*, *OWL / RDF Files*, *UML 1.4 Class Diagrams*, *Protégé XMI Files* e *RDF Files*), bem como criá-los a partir de arquivos existentes, uma vez que a sintaxe dos arquivos de entrada estejam corretas. Na exportação do modelo construído, disponibiliza as opções “*save as*” (no formato RDF/XML), “*export to format*” (nos formatos CLIPS, HTML, N-TRIPLE, N3, OWL, RDF Schema, RDFS Class Hierarchy, TURTLE) e “*Convert Project to Format*”, para qualquer dos tipos de projetos relacionados na criação. O nome do *plug-in* responsável pelo padrão MOF é *XMI Backend* que permite a interoperabilidade do metamodelo Protégé com outras ferramentas.

Text2Onto permite importar e exportar nos formatos OWL, RDFS e *KAON-OI Models*.

Dois *frameworks* de código aberto podem ser utilizados para facilitar a persistência de modelos armazenados no formato XMI: MDR (*Metadata Repository*) do Projeto Netbeans da *Sun Microsystems* e o EMF (*Eclipse Modelling Framework*), do Projeto Eclipse, da IBM. Ambos tratam os metadados no padrão MOF e possuem mecanismos elaborados para edição e transformação dos arquivos XMI.

⁷⁵ <http://umt-qvt.sourceforge.net/>

⁷⁶ <http://www.modelware-ist.org/>

5.1.3 Realizar Aprendizado de Ontologias

Esta atividade foi realizada utilizando recursos dos sistemas educacionais selecionados, agrupados em *schematas* - modelo de dados do Teleduc, modelo de dados do COL e diagrama de classes do IMS LD - e em textos – especificação do IMSL LD.

5.1.3.1 Aprendizado de Ontologias a partir de *schematas*

Os administradores dos sistemas educacionais Teleduc e COL disponibilizaram os *scripts* de criação de suas bases de dados para a realização do processo Engenharia Reversa (1e) do *framework* proposto, conforme representado na Figura 12. Os metadados dos SGBDRs (1c) dos sistemas existentes (1a), sendo SQL Server no caso do COL e MySQL no caso do Teleduc, serviram de entrada para a ferramenta CASE MagicDraw, versão 8.0, com a qual foi realizado o processo de Engenharia Reversa (1e), atualizando o Repositório (1d).

a) Modelo de Dados do COL

Para um melhor entendimento da execução dos processos e do poder da utilização dos padrões com as ferramentas descritas, foram selecionadas três tabelas, de um total de 600, do Sistema COL: "Disciplinas", "Modulo" e "Composicao". A Figura 16 apresenta um subconjunto do *script* SQL ANSI de criação destas três tabelas.

```

CREATE TABLE Disciplinas (
    id int NOT NULL ,
    codigo char (8) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL ,
    nome varchar (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL ,
    descricao text COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    professor int NULL ,
    link_glossario varchar (128) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    link_bibliografia varchar (128) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    link_sumario varchar (128) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    link_extra_nome varchar (40) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    link_extra varchar (128) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    grupo_padrao int NULL ,
    versao char (40) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON PRIMARY TEXTIMAGE_ON PRIMARY;

CREATE TABLE Modulos (
    id int NOT NULL ,
    nome varchar (50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL ,
    descricao text COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    url varchar (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    professor int NULL ,
    teste bit NULL ,
    subdiretorio char (255) COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON PRIMARY TEXTIMAGE_ON PRIMARY;

CREATE TABLE Composicao (
    disciplina int NOT NULL ,
    modulo int NOT NULL ,
    ordem int NULL ,
    separador char (1) COLLATE SQL_Latin1_General_CP1_CI_AS NULL ,
    texto text COLLATE SQL_Latin1_General_CP1_CI_AS NULL
) ON PRIMARY TEXTIMAGE_ON PRIMARY;

ALTER TABLE Composicao ADD
    CONSTRAINT FK_Composicao_Disciplinas FOREIGN KEY (disciplina) REFERENCES Disciplinas (id),
    CONSTRAINT FK_Composicao_Modulos FOREIGN KEY(modulo) REFERENCES Modulos (id);

```

Figura 16 - Subconjunto do *script* de criação SQL, obtido da base de dados CoL

Através deste *script*, foi realizada a Engenharia Reversa (1e) com a ferramenta MagicDraw com a seguinte seqüência de escolhas no seu menu: *Tools, Quick Reverse, Quick Reverse DDL, Microsoft SQL Server Dialect e Reverse from files*. Ao realizar a atividade com sucesso, a ferramenta mostrou um pacote contendo classes, atributos, operações e relações de dependência, conforme mostra a Figura 17. Os retângulos indicam as classes (de acordo com a notação UML), tendo na parte superior o nome da classe, no centro os atributos com nomes, tipos e restrição para valores nulos, e na parte inferior as operações. As setas entre as classes indicam relações de dependência recuperadas, ou seja, uma situação na qual uma mudança no elemento destino requer uma mudança no elemento de origem na dependência. Todos os elementos do modelo Relacional presentes no *script* ficaram representados no modelo de classes. Por exemplo, a tabela “Modulos”, com suas colunas (id, nome, descricao, url, professor, teste e subdiretorio) e sua chave estrangeira com a tabela “Composicao” (FK_Composicao_Modulos), transformou-se na classe “Modulos”, com os atributos (id, nome, descricao, url, professor, teste e subdiretorio) e com uma associação de dependência com a classe “Composicao”. A associação é estabelecida através do atributo “id” da classe “Modulo” denominado “modulo” na classe “Composicao”. O mesmo tipo de relação ocorre com a classe “Disciplinas”.

Figura 17 - Subconjunto do diagrama de classes UML, obtido a partir do *script* de criação SQL

Neste caso a Extração de Instâncias (M0) dos Modelos de Dados (M1) - 2a - é uma funcionalidade da ferramenta MagicDraw realizada com a seguinte seqüência de escolhas no seu menu: *File, Export, EMF ULM2 (V1.x) XMI*. Assim a partir deste modelo, sem nenhuma alteração, extraíram-se as instâncias dos modelos de dados (2a) obtendo sua representação no padrão MOF/XMI (2a1). Alguns fragmentos do arquivo no formato padrão XMI obtido podem ser observados na Figura 18, com destaque para os elementos do modelo referentes à classe “Modulos”.

```

<ownedMember xmi:type="uml:Class" xmi:id="_15_1EAPbeta_a9c02c5_1203510561187_914413_599" name="Modulos">
  <ownedAttribute xmi:id="_15_1EAPbeta_a9c02c5_1203510561187_856877_602" name="id" visibility="private">
    <type xmi:type="uml:PrimitiveType" href="Generic_DDL_Profile.profile.uml2#_91a0295_1073488272109_297666_31"/>
  </ownedAttribute>
  <ownedAttribute xmi:id="_15_1EAPbeta_a9c02c5_1203510561187_652197_604" name="nome" visibility="private">
    <type xmi:type="uml:PrimitiveType" href="Generic_DDL_Profile.profile.uml2#_91a0295_1073488395843_82199_36"/>
  </ownedAttribute>
  <ownedAttribute xmi:id="_15_1EAPbeta_a9c02c5_1203510561187_673925_608" name="descricao" visibility="private">
    <type xmi:type="uml:PrimitiveType" href="Generic_DDL_Profile.profile.uml2#_91a0295_1073552990296_244209_17"/>
  </ownedAttribute>
  <ownedAttribute xmi:id="_15_1EAPbeta_a9c02c5_1203510561187_224326_610" name="url" visibility="private">
    <type xmi:type="uml:PrimitiveType" href="Generic_DDL_Profile.profile.uml2#_91a0295_1073488395843_82199_36"/>
  </ownedAttribute>
  <ownedAttribute xmi:id="_15_1EAPbeta_a9c02c5_1203510561187_453860_614" name="professor" visibility="private">
    <type xmi:type="uml:PrimitiveType" href="Generic_DDL_Profile.profile.uml2#_91a0295_1073488272109_297666_31"/>
  </ownedAttribute>
  <ownedAttribute xmi:id="_15_1EAPbeta_a9c02c5_1203510561203_890351_616" name="teste" visibility="private">
    <type xmi:type="uml:PrimitiveType" href="Generic_DDL_Profile.profile.uml2#_91a0295_1073552685796_893027_3"/>
    <defaultValue xmi:type="uml:OpaqueExpression" xmi:id="_15_1EAPbeta_a9c02c5_1203510561218_519981_665" body="(0)"/>
  </ownedAttribute>
  <ownedAttribute xmi:id="_15_1EAPbeta_a9c02c5_1203510561203_812680_618" name="subdiretorio" visibility="private">
    <type xmi:type="uml:PrimitiveType" href="Generic_DDL_Profile.profile.uml2#_91a0295_1073487955000_700251_12"/>
  </ownedAttribute>
</ownedMember>

...
<ownedMember xmi:type="uml:Dependency" xmi:id="_15_1EAPbeta_a9c02c5_1203510561234_847556_679" name="FK_Composicao_Modulos"
1EAPbeta_a9c02c5_1203510561203_28833_622" supplier="_15_1EAPbeta_a9c02c5_1203510561187_914413_599">
  </ownedMember>
...

```

Figura 18 - Subconjunto do arquivo XML, obtido do diagrama UML

Na seqüência, com o processo de Transformação de Metamodelos (2c) e as regras para transformação (2a2), obteve-se um metadado no modelo OWL (2e). Este passo foi executado já no ambiente de ontologias com a ferramenta Protégé com a seguinte seqüência de escolhas: *File, New Project, UML 1.4 Class Diagram* e *Create from Existing Sources*.

b) Modelo de Dados do Teleduc

A mesma seqüência de execução dos processos foi realizada com os *scripts* do sistema Teleduc e todos os elementos do Modelo Relacional ficaram representados no modelo de classes obtido (Figura 19). Por exemplo, a tabela “Atividade_topicos”, com suas colunas (cod_topico, cod_topico_pai, cod_usuario, topico, tipo_compartilhamento, data e posição_topico) e sua chave estrangeira com a tabela “Portfolio_topicos”, transformou-se na classe “Atividade_topicos”, com os atributos (cod_topico, cod_topico_pai, cod_usuario, topico, tipo_compartilhamento, data e posição_topico) e com uma associação de dependência com a classe “Portfolio_topicos”. A associação é estabelecida através dos atributos “cod_topico”, “cod_topico_pai” e “cod_usuario”.

Figura 19 – Modelo de Dados - Teleduc

A recuperação dos modelos mostrou o quanto sistemas aparentemente com mesmas funcionalidades como o COL e o Teleduc podem ter representações distintas principalmente nos conceitos adotados.

c) Diagrama de Classes UML do IMS LD

O modelo do IMS LD encontra-se representado e disponível no metamodelo da ferramenta CASE Poseidon⁷⁷ e é formado por um conjunto de diagramas de classes, descrições e propriedades das classes e associações entre elas. A versão utilizada da ferramenta foi “Poseidon for UML Community Edition 6.0” e uma vez realizada a importação do arquivo XMI disponibilizado, o Repositório (1d) ficou com aproximadamente 40 diagramas de classes. Os diagramas UML visualizados na ferramenta Poseidon são idênticos ao modelo conceitual do IMS LD – Figura 20 -, e detalhados no Anexo D.

Figura 20 - Modelo Conceitual do IMS LD (IMS, 2007)

A Extração de Instâncias (M0) dos Diagramas UML (M1) (2b), na Figura 12, utilizou a funcionalidade da ferramenta Poseidon para exportar modelos no formato XMI e foi realizada sem nenhuma intervenção manual nos diagramas, resultando no Metadado XMI (2b1). A seqüência de escolhas no menu foi: *File* e *Export Project for XMI*.

A Transformação entre Metamodelos (2c), na Figura 12, foi realizada com a ferramenta CIMTool. CIMTool possui as regras R2 (2b2) definidas internamente e

⁷⁷ <http://dspace.ou.nl/handle/1820/659>

gerou o Metadado OWL (2e). Poseidon produziu um arquivo XML com nível M2 e M3 e CIMTool transforma M2 da UML para M2 da OWL, ou seja, as duas ferramentas seguem o padrão MOF para os metamodelos considerados.

Para efeitos de teste, esta transformação foi realizada também com as ferramentas Xalan e ATL.

5.1.3.2 Aprendizado de Ontologias a partir de textos

Para o processo Extração dos Elementos da Ontologia (3a) foi utilizada a ferramenta Text2Onto, versão 1.2b. A Figura 21 mostra os algoritmos disponibilizados para extração, separados por elemento aprendido: conceitos (*Concept*), instâncias (*Instance* e *InstanceOf*), sinônimos (*Similarity*), relação taxonômica (*SubClassOf*), outras relações (*Relation*) e (*SubtopicOf*). Por exemplo, para aprender as relações taxonômicas presentes no texto, existe o algoritmo denominado *PatternConceptClassification*.

Figura 21 – Algoritmos da ferramenta Text2Onto

Os algoritmos foram executados incrementalmente sobre as páginas HTML do Modelo de Informação do IMS LD e segundo a ordem sugerida graficamente pela ferramenta. A ferramenta permite dispor os elementos aprendidos resultantes de cada algoritmo em sub-pastas (*Concept*, *Instance*, *Similarity*, *SubclassOf*, *InstanceOf*, *Relation*, *SubtopicOf*), sempre apresentando o *Label* para o elemento aprendido com sua relevância (*Relevance*). Após as execuções, observação da relevância dos elementos aprendidos, e sem nenhuma intervenção manual, foi realizada a Extração dos Elementos da Ontologia (3a) no Metadado (3b) na própria ferramenta Text2Onto. Esta opção exporta o resultado obtido no formato OWL, o que corresponderia, a princípio, à Transformação entre Metamodelos (3c) com as regras R3 (3b1) obtendo o Metadado OWL (3d).

As ferramentas para aprendizado a partir de textos estão preparadas para analisar uma quantidade maior de documentos. Nesta prova de conceito foi utilizado somente um texto.

5.1.4 Utilizar Ambiente de Ontologias

A ferramenta Protégé dispõe das funcionalidades para compor o bloco Ambiente de Ontologias e, principalmente, realizar o Tratamento de Ontologias (4a) a partir dos diversos Metadados OWL (2e) e (3d) obtidos, de tal forma a disponibilizar a Representação Inicial da Ontologia (4b) numa Base Ontológica (5b).

A utilização do ambiente ocorreu com o uso das funcionalidades para criar novos projetos a partir de arquivos existentes, editar a representação inicial da ontologia, obter métricas do modelo, verificar a expressividade OWL DL, validar o modelo OWL e visualizar a representação, entre outras.

A edição da representação inicial da ontologia contou com a busca e seleção dos elementos de correspondência entre metamodelos enumerados nos Quadros 16, 17 e 18. O uso de mnemônicos significativos dos sistemas existentes auxilia muito no aprendizado do conhecimento representado. Isso pode ser observado pelos nomes das tabelas (por exemplo: "Disciplinas") ou pelos nomes de colunas (por exemplo: nome, código, versão, entre outras).

As métricas da representação e sua expressividade podem ser obtidas com a opção *Project, Metrics*. Um resumo com a quantidade de elementos presentes em cada representação inicial pode ser visualizado.

A validação da representação obtida pode ser executada com a opção *Tools, WonderWeb Validator*. Esta opção ativa a ferramenta *OWL Validator*, também conhecida como *vOwLidator*⁷⁸ na Web, retornando um relatório com resumo da análise realizada para cada modelo obtido e a classificação da representação em *OWL Lite* ou *OWL DL*.

A visualização da ontologia ocorre ao ativar o *plug-in* denominado Jambalaya no Protégé. Para fazer isto é necessário selecionar, no menu principal do Protégé, a opção *Project, Configure, Tab Widgets, JambalayaTab*. Ao final da seleção uma nova janela denominada Jambalaya com todas as opções de visualização fica disponível no ambiente Protégé.

5.1.5 Selecionar Método para Engenharia Progressiva

Para a Engenharia Progressiva é necessário selecionar o Método para Desenvolvimento (5a) mais adequado para detalhar a Representação Inicial da Ontologia (4b), armazenada na Base Ontológica (5b), até sua Representação Final. Muitas metodologias são candidatas diante do contexto do *framework* proposto como Methontology, UPON e Diligent, conforme Quadros 2 e 3.

No caso das representações iniciais obtidas a partir do IMS LD, o desenvolvimento descentralizado e colaborativo é adequado por se tratar da criação de uma ontologia de domínio e já possuir metadados padronizados e compartilhados entre os participantes da comunidade criadora do metadado. Isso traz a interoperabilidade desejada entre as aplicações do domínio da Educação, através da exposição da semântica dos metadados, e já representa o consenso dos usuários da comunidade. Sua construção incremental e colaborativa permite a evolução da ontologia, sem a perda da referência (o metadado), considera a existência de várias representações iniciais independentes e unifica as representações numa ontologia de domínio.

Neste contexto foi selecionado o método Diligent por abordar a Engenharia de Ontologia como um processo social envolvendo muitos participantes – engenheiros

⁷⁸ <http://www.mygrid.org.uk/OWL/Validator>

do conhecimento, especialistas do domínio e usuários, presentes em muitas localizações (TEMPICH et al., 2005). Considera cinco processos: construção, adaptação local, análise, revisão e manutenção local.

A construção inicia com os usuários do domínio (os especialistas, os engenheiros do conhecimento e os engenheiros de ontologias) construindo suas representações iniciais da ontologia. Neste momento, eles podem ter o suporte do *framework* proposto. Esta equipe, com diferentes abordagens e localização geográfica, busca a construção de uma primeira versão - pequena, consensual e compartilhada - da ontologia, porém não requer perfeição na totalidade de representação com respeito ao domínio.

Quando a representação da ontologia estiver disponível, os usuários realizam uma adaptação na ontologia local para utilizá-la de acordo com seus propósitos. Tipicamente, novos requisitos de negócio, ou mudanças organizacionais ou dos usuários, envolvem as ontologias locais onde elas ficam livres para mudanças. Porém, quando a ontologia é compartilhada entre os usuários de uma comunidade, elas não podem ser diretamente alteradas.

O processo de análise administra a ontologia compartilhada através do controle dos requisitos de mudanças, identificação de similaridades entre as ontologias locais, e validação da nova versão da ontologia compartilhada. Assim, nem todas as mudanças introduzidas ou requisitadas pelos usuários serão representadas e disponibilizadas na ontologia compartilhada.

O processo de revisão da ontologia compartilhada busca elucidar as divergências entre a ontologia local e a ontologia compartilhada procurando balancear e representar as diferentes abordagens envolvidas no processo.

Quando uma nova versão da ontologia compartilhada é liberada, os usuários alteram suas ontologias locais para melhor usar o conhecimento representado na nova versão, realizando assim o processo de manutenção local.

A utilização desta metodologia no domínio da Educação é extremamente adequada quando se considera a utilização de metadados padronizados. Os metadados têm uma ontologia compartilhada, definida e mantida descentralizada e colaborativamente, e os diferentes fornecedores de sistemas educacionais têm sua ontologia local, adaptada de acordo com suas funcionalidades, mas compatível e alinhada com a ontologia compartilhada.

O método Diligent foi selecionado, porém não foi contemplado neste trabalho, principalmente, pelo tempo necessário para sua completa realização e análise.

5.2 Resultados Obtidos

Os resultados obtidos foram separados em categorias para facilitar a discussão: quantidade de elementos aprendidos, expressividade da representação OWL e conhecimento do engenheiro de ontologias.

5.2.1 Quantidade de Elementos Aprendidos

Os elementos obtidos são representativos no domínio da Educação e significativos, tanto na quantidade, quanto na consistência sintática da representação inicial obtida. A Tabela 1 mostra a quantidade de classes, propriedades e instâncias presentes em cada representação inicial obtida.

Tabela 1 – Quantidade de Elementos Aprendidos

Elementos	COL	Teleduc	IMS LD	Texto
Classes nomeadas	9	22	193	9
Propriedades	9	30	368	2
Instâncias dos elementos aprendidos				
<i>Concept</i>	-	-	-	1054
<i>InstanceOf</i>	-	-	-	419
<i>Relation</i>	-	-	-	11
<i>Subclassof</i>	-	-	-	47

A prova de conceito tratou de poucos elementos no caso dos *schematas*, parecendo à primeira vista que uma reengenharia manual seria mais fácil de ser realizada. Porém, ao considerar o número total de tabelas (600 no caso do COL), a reengenharia manual não será trivial, mostrando que a utilização do processo descrito é mais eficiente.

A representação inicial da ontologia para IMS LD traz 193 classes e 368 propriedades do modelo IMS LD; entre elas destacam-se “*Activity-structure*” com sua definição, suas propriedades e restrições apresentadas na Figura 22 lado direito, e

Amorim, Lama e Vila (2006) construíram manualmente uma ontologia para representar a semântica do IMS LD, nível A, utilizando Protégé-OWL. Definiram uma taxonomia de conceitos, na qual as relações entre os conceitos são explicitamente representados, bem como um conjunto de axiomas para expressar as restrições identificadas na análise do Modelo de Informação do IMS LD. Esta ontologia resultou em 35 classes, aproximadamente 200 propriedades e 20 axiomas e é utilizada no projeto EUME⁷⁹. A classe *Activity-structure* com sua definição, suas propriedades e restrições pode ser vista na Figura 23.

⁷⁹ <http://www-gsi.dec.usc.es/~eume/>

Figura 23 – Classe “Activity-structure” do Projeto EUME

Todas as classes existentes na ontologia do projeto EUME existem na representação inicial aprendida embora ocorram pequenas alterações na nomenclatura no projeto EUME – “*Activity*” versus “*Activities*”, “*Complete-Unit*” versus “*Complete-unit-of-learning*” – porém os comentários são idênticos.

As propriedades existentes na ontologia do projeto EUME possuem as mesmas propriedades da representação inicial aprendida adicionando novas. As propriedades adicionais são “*execution-entity-ref*”, “*execution-order*”, “*inverse-of-component-ref*”, “*inverse-of-execution-entity-ref*” e “*number-to-select*”, as demais propriedades são as mesmas. Isto pode ser observado com a classe “*Activity-structure*” nas Figuras 21 e 22.

Com a utilização do *framework* proposto, o engenheiro de ontologia consumiria menos tempo para criação da representação inicial da ontologia, principalmente, na aquisição e definição dos conceitos, propriedades e relações taxonômicas presentes nos modelos, conceitual e de informação, do IMS LD que não precisam serem definidas manualmente e não passam por nenhuma interpretação pessoal do engenheiro de ontologia. A ontologia aprendida com o *framework* proposto apresenta 193 classes contra 35 definidas manualmente no projeto EUME,

O tempo ganho poderia ser dedicado à construção de axiomas e à seleção dos conceitos significativos para o nível A do IMS LD. Além disto, os engenheiros de ontologia têm a certeza da sintaxe correta da representação aprendida e da correspondência entre o metadado padrão e a representação inicial da ontologia, embora a inserção de relações e axiomas ainda gere trabalho manual como a inserção da propriedade “*execution-entity-ref*”, mostrada na Figura 24 parte inferior, com suas características e operadores disponíveis.

Figura 24 – Edição da propriedade “*execution-entity-ref*”

No caso da representação inicial do IMS LD obtida a partir de textos, ocorrem instâncias de conceitos e relações. O engenheiro de ontologias precisará ainda transformá-las em classes do domínio, utilizando a funcionalidade do Protégé denominada “*Convert individual to class*”. Isto ocorre com a seleção individual de conceitos significativos e pode ser útil devido ao elevado número de conceitos aprendidos.

5.2.2 Expressividade da representação OWL

O poder de expressividade da representação da ontologia obtida em OWL é determinado pelos operadores (*constructs*) e propriedades das classes suportados e pelos tipos de axiomas presentes na representação.

A versão OWL *Lite* é equivalente à lógica de descrição *SHIF(D)*, caracterizada pelos operadores para hierarquias de classificação e restrições simples como

cardinalidade máxima e mínima com valores 0 e 1. As propriedades podem ser transitivas, simétricas, funcionais e funcionais inversas.

A versão OWL *DL* é equivalente à lógica de descrição *SHOIN(D)*, capaz de representar operadores da família da lógica de descrição ALC - negação, intersecção, quantificadores universal e existencial $\exists$, mais hierarquias de papéis, propriedade transitiva, inversa, restrições de cardinalidade não qualificadas, enumeração, os operadores $\leq_n$, $\geq_n$ e hierarquia de papéis, ou seja, admite o conectivo $\subseteq$ entre papéis. Protégé implementou algoritmos para verificação da expressividade segundo *SHIF(D)* e *SHOIN(D)*, apresentando letras, conforme os operadores utilizados na definição dos conceitos, mostradas no Quadro 28.

Letra	Descrição
<i>AL</i>	- União ($\sqcup$), intersecção ($\sqcap$), negação ($\neg$), quantificador existencial ($\exists$) e quantificador universal ($\forall$);
<i>ALO</i>	- Intersecção ($\sqcap$) e quantificador universal ($\forall$);
<i>AL(-1)</i>	- Intersecção ($\sqcap$), quantificador universal ($\forall$) e negação ($\neg$);
<i>C</i>	- Negação complexa como a expressão “A or B” ;
<i>U</i>	- União ($\sqcup$);
<i>E</i>	- Todos os quantificadores existenciais ($\exists$);
<i>N</i>	- Restrições numéricas (restrições cardinalidade e também inclui propriedades funcionais);
<i>Q</i>	- Número de restrições de cardinalidade qualificadas;
<i>H</i>	- Hierarquia de papéis (propriedades);
<i>I</i>	- Papéis inversos (propriedades inversas ou simétricas);
<i>S</i>	- <i>AL C</i> com papéis transitivos;
<i>O</i>	- Nomeadas como <i>one of</i> e <i>hasValue</i> ;
<i>F</i>	- Propriedades funcionais;
(D)	- Tipos de dados.

Quadro 28 - Expressividade da representação OWL

O resumo do poder de expressividade de cada representação obtida segundo os algoritmos aplicados no Protégé pode ser visto na Tabela 2.

Tabela 2 – Expressividade das Representações

Recurso de Entrada	Expressividade DL
COL – Modelo de Dados	$\mathcal{ALI}$
Teleduc – Modelo de dados	$\mathcal{ALIF}$
IMS LD – Diagramas UML	$\mathcal{ALF}(\mathcal{D})$
IMS LD – Textos	$\mathcal{AL}$

Todas as representações iniciais aprendidas são $\mathcal{AL}$.

No caso da representação inicial obtida do COL ocorre a presença de propriedades inversas (I). Todas as tabelas foram definidas como classes, todas as colunas foram definidas como propriedades do domínio, todas as chaves estrangeiras foram definidas como propriedade inversa de objeto com a chave primaria da tabela. Um exemplo é a classe “Composicao” escrita assim em OWL, conforme mostrada a seguir:

```

<owl:Class rdf:about="http://...#Composicao"> ...
</owl:Class>
<owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="http://...#Composicao.id">
    <owl:inverseOf>
  <owl:ObjectProperty rdf:about=" http://...#Disciplinas.disciplina"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="http:...#Disciplinas"/>
    <owl:inverseOf>
      <owl:ObjectProperty rdf:about="http:...#Modulos.modulo"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="http:...#Modulos"/> ...
  <rdfs:Property rdf:about="http://...#Composicao.modulo">
    ...
    <rdfs:domain rdf:resource="http://...#Composicao"/> ...
  </rdfs:Property>
  <rdfs:Property rdf:about="http://...#Composicao.disciplina">
    <rdfs:domain rdf:resource="http:...#Composicao"/>
    ...
  </rdfs:Property>
  <rdfs:Property rdf:about="http://... #Composicao.ordem">
    <rdfs:domain rdf:resource="http://...#Composicao"/>
    ...
  </rdfs:Property>
  <rdfs:Property rdf:about="http...#Composicao.separador">
    ...
    <rdfs:domain rdf:resource="http://...#Composicao"/> ...
  </rdfs:Property>
  <rdfs:Property rdf:about="http://...#Composicao.texto">
    <rdfs:domain rdf:resource="http://...#Composicao"/>
    ...
  </rdfs:Property>

```

As representações obtidas do Teleduc e IMS LD possuem propriedades funcionais inversas ($\mathcal{F}$) resultantes dos atributos referentes às relações existentes nos diagramas de classes de origem. Como ocorre com a propriedade “Obrigatoria_itens.cod_item” escrita assim em OWL:

```
<owl:ObjectProperty rdf:about="http://...#Obrigatoria_itens.cod_item">
  <rdfs:range rdf:resource="http://...#Obrigatoria_itens_enderecos"/>
  <rdf:type rdf:resource="http://...#FunctionalProperty"/>
  ...
</rdf:Property>
```

Apesar da semântica de cada um dos modelos – dados, UML e ontologia - ser diferente, o conhecimento representado nos recursos selecionados não é perdido, sintaticamente, nas transformações realizadas.

O conhecimento das funcionalidades existentes no ambiente de ontologias, como visualização parcial e validação de ontologia, e das abstrações presente em cada modelo facilita o estudo e análise dos elementos aprendidos além de promover o aperfeiçoamento da representação final.

5.2.3 Conhecimento do Engenheiro de Ontologias

A primeira preocupação na elaboração do *framework* proposto teve a ver com a dificuldade dos engenheiros de ontologia em modelar ontologias, devido ao desconhecimento do domínio ou da sintaxe de representação da ontologia. Assim, o *framework* serve a dois propósitos. Em primeiro lugar, transforma representações existentes numa terminologia adequada ao contexto do domínio. Em segundo lugar, a representação obtida está sintaticamente correta, podendo posteriormente ser enriquecida com outros elementos mais complexos como axiomas ou propriedades de classes.

O engenheiro de ontologias visualiza facilmente a lista de classes, propriedades e restrições existentes na representação obtida e para cada elemento pode ler os comentários recuperados que detalham o mnemônico utilizado na representação.

Ao observar a lista das classes e propriedades da representação obtida para o COL, o engenheiro de ontologias tem acesso à classe “Turmas” e suas propriedades como

“avaliacao”, “ementa”, “descricao”, “data_ini”, “data_fim”, “data_vig_ini”, “data_vig_fim”, entre outras. Ao examinar as propriedades inversas descobre as relações entre as classes, como com a classe “Composicao” relacionada com as classes “Disciplinas” e “Modulos”.

Figura 25 – Classes e Propriedades da representação inicial obtida para o COL

Ao ler o comentário da classe “*Activity_structure*” da representação obtida do IMS LD o engenheiro de ontologias obtém uma definição da classe: “*An activity-structure groups activities in sequences or selections. The tree is handled depth first (and not breadth first).*” (Figura 22). A redação em inglês obriga a tradução por parte do engenheiro de ontologias e isto pode ser mais uma dificuldade a ser vencida na conceitualização de ontologias de domínio.

Assim, o engenheiro de ontologias tem acesso no mínimo à terminologia utilizada no domínio e reduz a necessidade de um conhecimento anterior especializado do domínio. A representação provoca também dúvidas, questionamentos, ambigüidades e comparações sobre o domínio a serem esclarecidos com o especialista do domínio. Isto tudo deixa o engenheiro de ontologias melhor preparado para participar do processo de Construção do método Diligent, ou da Análise do Domínio na engenharia de ontologias tradicional, ou das fases Aparar e Refinar do projeto KAON, onde os elementos principais da ontologia estão sendo definidos.

5.2.4 Diferenças nas representações aprendidas

Ontologias são especificações complexas e a existência de diferentes tipos leva-as a terem diferentes abrangências no mesmo domínio, ou seja, diferentes ontologias podem ser consideradas corretas e apropriadas para representar o conhecimento nele contido.

Diferentes sistemas educacionais apresentam diferentes metadados e resultam conseqüentemente em diferentes representações iniciais obtidas a partir do *framework* proposto. No caso do COL e do Teleduc, muito embora tratem do mesmo domínio e apresentem mesmas funcionalidades, os elementos aprendidos são totalmente distintos. Um exemplo é o atributo “professor” presente na entidade “Disciplinas” do modelo COL e ausente no modelo Teleduc. As representações iniciais aprendidas podem servir para a construção das respectivas ontologias de aplicação com as definições necessárias para modelar o conhecimento representado em cada sistema.

5.3 Discussão

A prova de conceito do *framework* proposto mostrou a simplicidade para aplicação dos padrões da OMG e W3C uma vez conhecida a correspondência entre os modelos de origem e de destino e a existência de muitas ferramentas atendendo aos requisitos para participação do *framework*.

Os especialistas em ontologia ainda não chegaram num consenso sobre um metamodelo padrão para sistematizar a construção de ontologias, comprovado sua falta nas ferramentas para AO a partir de textos e para o ambiente de ontologias. A Engenharia de Software com o ODM pode servir como base na busca por este metamodelo padrão uma vez que já produz resultados satisfatórios como o obtido com o *framework* proposto.

A maioria das ferramentas é livre, resulta de trabalhos de pesquisas, tem interface gráfica amigável e possui tutoriais disponíveis permitindo sua fácil e rápida utilização.

Uma vez selecionadas as ferramentas, resta ao engenheiro de ontologias dedicar-se à seleção dos recursos do domínio e ao entendimento dos resultados obtidos do ponto de vista do domínio e da linguagem representada para aplicação na engenharia progressiva. O engenheiro de ontologias tem acesso à representação OWL e, mesmo sem dominar sua sintaxe, pode enriquecê-la com o acréscimo de novos elementos no Ambiente de Ontologias. O conhecimento das funcionalidades existentes neste ambiente, como visualização parcial e validação de Ontologia, e das abstrações presentes em cada modelo facilita o estudo e análise dos elementos aprendidos, além de promover o aperfeiçoamento da representação final. Assim as representações de conhecimento modeladas em sistemas existentes não são perdidas e podem ser aproveitadas no universo da Web Semântica.

6 CONSIDERAÇÕES FINAIS

O sucesso da Web Semântica depende da disponibilidade de ontologias. A Engenharia de Ontologias não é uma atividade trivial e requer tempo, disponibilidade e consenso dos especialistas, pois a semântica dos termos pode variar de um contexto para outro, de um local para outro, e de uma pessoa para outra. Este trabalho propôs um *framework* para obtenção de representações iniciais de ontologias, aprendidas a partir da transformação de representações presentes em modelos de sistemas existentes, aplicando metamodelos padronizados pelo OMG e W3C e utilizando ferramentas da Engenharia de Software e Engenharia de Ontologias já construídas. A seguir apresenta-se uma síntese do trabalho e sugestões para trabalhos futuros.

6.1 Síntese do Trabalho

A Web Semântica tem desencadeado o uso de técnicas, linguagens, ferramentas e metodologias cada vez mais especializadas para promover a busca, verificação, recuperação e análise de recursos independente da interferência humana. Assim, novas camadas de informações são disponibilizadas na Web, agregando, além de informações que representam suas características léxicas e sintáticas, as informações com as características semânticas presentes em metadados, anotações e/ou ontologias.

Para diminuir a complexidade em torno da Web Semântica, organismos internacionais como OMG e W3C criaram arquiteturas padronizadas para sua concepção – MDA e ODM - e implementação – RDF, OWL e SPARQL -, respectivamente. A união dessas arquiteturas permite o compartilhamento de modelos e conceitos entre métodos e ferramentas diferentes. Isso faz com que o trabalho do engenheiro de ontologias seja dividido entre entender o domínio e os recursos da Engenharia de Ontologia para representação das ontologias como metodologias com abordagem centralizada ou colaborativa e entender o funcionamento das ferramentas para edição, avaliação e consultas das ontologias.

Isto tudo torna a construção de ontologias um processo oneroso e demorado trazendo dificuldades para a expansão da Web Semântica.

O engenheiro de ontologias pode utilizar sistemas de aprendizagem como o ASIUM, Text2onto e Eronto para realizar o Aprendizado de Ontologias – um conjunto de métodos e técnicas usado para construir uma ontologia através de processos automáticos ou semi-automáticos aplicados em recursos existentes num domínio específico. Esses sistemas estão associados ao elemento a ser aprendido – conceito, relação, função e axioma - e a sofisticadas técnicas do PLN e Aprendizagem de Máquinas.

Neste contexto, onde os engenheiros de ontologias precisam lidar com recursos provenientes de diversas disciplinas, o *framework* proposto permite unificar resultados existentes, sejam das pesquisas com Aprendizado de Ontologias, da Engenharia de Software ou dos sistemas já existentes. Isto é possível pelo compartilhamento de um metamodelo comum para representação da ontologia – o ODM para OWL – que garante a interoperabilidade entre as diversas ferramentas utilizadas, pela correspondência dos elementos entre metamodelos conhecidos – modelo de dados, diagrama de classes da UML e textos - e o modelo OWL, pela definição dos requisitos para as ferramentas participarem do *framework* e pelo detalhamento das atividades a serem realizadas pelo engenheiro de ontologias na utilização do *framework* proposto.

Assim, o *framework* proposto faz uso do Aprendizado de Ontologias a partir de metamodelos padrões do OMG, considera modelos existentes e, apesar da semântica de cada um deles ser diferente, realiza a transformação sintática do conhecimento representado.

O *framework* proposto simplifica o processo de criação de representações iniciais de ontologias e estabelece um ponto de partida importante para o trabalho do engenheiro de ontologias. Mesmo com a necessidade de execução da engenharia progressiva, onde é preciso validar e completar a representação da ontologia, o uso deste *framework* pode reduzir consideravelmente o esforço com a aquisição de conceitos e relações presentes no domínio, bem como na sua representação sintática já correta. Este apoio ao trabalho do engenheiro de ontologias contribui para diminuir o tempo total necessário para a construção final da ontologia e para melhorar o uso e aplicação de ontologias. As representações iniciais aprendidas serão o ponto de convergência dos engenheiros de ontologia, sempre discutindo o

conhecimento representado e a descrição dos elementos mais importantes, principalmente nas atividades para obter o consenso dos especialistas do domínio.

A prova de conceito do *framework* proposto mostrou a viabilidade do uso de padrões de metamodelos nas ferramentas para aprendizado e tratamento de ontologias. Os metadados existentes em outras ferramentas, não voltadas ao desenvolvimento de ontologias, podem ser reutilizados em ferramentas voltadas para representação de ontologias, desde que seus metamodelos sejam conhecidos ou sigam os padrões sugeridos por OMG e W3C. Alguns aspectos devem ser considerados como os metamodelos para *schematas*, para as ferramentas para a Engenharia de Ontologias e a versão das especificações consideradas.

Uma consideração importante quanto aos metamodelos para *schematas* é ter presente a existência de dois tipos de diagramas UML - aqueles construídos para representar um sistema qualquer e aqueles construídos para utilizar uma ontologia. Os primeiros não consideram os elementos da ontologia e precisam passar por um processo de transformação, onde alguns elementos não encontram correspondência direta como os axiomas. Já os segundos representam os elementos da ontologia com os recursos do ODM e OUP encontrando assim correspondência direta nos processos de transformação. Neste trabalho, somente os primeiros foram considerados.

As ferramentas para o AO e ambiente de ontologias, na sua grande maioria, têm metamodelos proprietários e muitas vezes apresentam-nos na forma de textos explicativos ou com modelos gráficos como Text2Onto. Estas ferramentas, por não possuírem um metamodelo padrão, precisam construir processos de transformação para outras representações como para o modelo OWL. A proposta do *framework* é inserir os padrões da Engenharia de Software, expressos pelo OMG (MDA, ODM, MOF e XMI), em todo universo da Engenharia de Ontologias desde o AO até o Ambiente de Ontologia. A ferramenta Protégé já está neste caminho ao disponibilizar seu metamodelo e realizar exportações no formato XMI.

Caso estas ferramentas não compartilhem um mesmo meta-metamodelo, como no caso dos *schematas*, é necessário investigar e detalhar o metamodelo de cada ferramenta. A simples representação final da ontologia em OWL não garante a expressividade dos elementos. Um conceito do domínio pode ser representado como uma instância na OWL e não como uma classe, por exemplo. Assim, para garantir os mapeamentos corretos, é necessário conhecer o metamodelo da

ferramenta e suas primitivas de representação. Muitas ferramentas já possuem mecanismos para exportação no formato OWL, mas é necessário utilizar o metamodelo padrão para garantir os níveis MOF de interoperabilidade entre ferramentas.

A versão das especificações consideradas é fundamental para os resultados finais obtidos. Tanto as ferramentas como as especificações dos padrões estão em constante evolução gerando e disponibilizando várias versões que nem sempre estão sincronizadas. Cada nova versão impõe novos elementos, tanto do metamodelo como dos modelos. No caso das especificações XMI e MOF existem as seguintes correspondências:

- XMI 1.1 corresponde ao MOF 1.3;
- XMI 1.2 e 1.3 corresponde ao MOF 1.4;
- XMI 2.0 corresponde ao MOF 1.4 com adição de *Schemas*;
- XMI 2.1 corresponde ao MOF 2.0.

Já o metamodelo OWL pode contemplar as diferentes versões OWL *Lite*, OWL *DL* ou OWL *Full*. E a UML também possui diferentes versões desde UML 1.0 até UML 2.1.1.

Algumas diferenças foram constatadas nos resultados obtidos em testes devido à falta de correspondência entre as versões OWL, UML, XMI e MOF. O uso de ferramentas com versões díspares traz diferenças nos elementos resultantes no formato XMI. A aderência ao padrão MOF observa-se pelos elementos OWL obtidos – elementos do domínio ou do metamodelo da ferramenta origem.

Ao utilizar os processos de transformações de metamodelos é importante observar as versões das especificações presentes nas ferramentas envolvidas. Encontrar ferramentas de transformação considerando versões mais atualizadas, tanto para o metamodelo de entrada como para o metamodelo de saída, pode não ser uma tarefa fácil. Por exemplo, não foi encontrada uma ferramenta para transformação para o metamodelo OWL *Full*. Principalmente, pela falta de ferramentas de AO de axiomas com metamodelo disponível e, também, pelos modelos de representação anteriores (de dados e de classes) não contemplarem a definição de axiomas, muito embora a maioria das representações atenderem ao metamodelo OWL *Lite*.

Os seguintes comentários são pertinentes em relação às ferramentas e tecnologia:

- as ontologias existentes, em sua grande maioria, têm uma versão OWL *Lite* e poucas evoluem para versões OWL *DL* e OWL *Full*. Isto se dá devido à

dificuldade de representação dos axiomas em ambientes que garantam execução dos mecanismos de raciocínio;

- as ferramentas CASE seguem UML e o padrão MOF e isto garante a interoperabilidade dos modelos entre diversas ferramentas inclusive com as ferramentas de implementação como SGBDR. O mesmo pode ser obtido com o ODM e OUP para modelar ontologias e OWL para implementá-las.
- o ambiente de ontologias no Protégé é completo e sua arquitetura aberta permite a existência de várias funcionalidades extras (*plug-ins*) para personalizá-lo e aperfeiçoá-lo.
- linguagens de transformações facilitam as transformações entre metamodelos com as folhas de estilo e regras de transformações definidas. Importante é tratar os mesmos níveis MOF entre os metamodelos considerados.
- a utilização de padrões pode agilizar outras áreas da Engenharia de Ontologias como avaliação e testes das ontologias construídas.

O domínio da Educação tem excelentes oportunidades para fazer uso da arquitetura oferecida pela WS, pois a Web já é uma fonte de referência para aprender sobre pessoas, lugares e assuntos diversos. Assim, a Web Semântica na Educação precisa abordar múltiplos conceitos como tipos de aprendizagem e ferramentas adequadas ao contexto de ensino, tratamento do conteúdo de aprendizagem baseado em objetos de aprendizagem e projeto instrucional, metadados padrões, serviços semânticos e personalização. Muitos projetos estão nesta direção como Tangram, SeLeNe, SKOS e TELOS. Neste trabalho a prova de conceito do *framework* proposto foi executada com os sistemas educacionais COL, Teleduc e Reload. Durante esta prova constatou-se que, no caso do AO a partir de *schematas*, o processo é muito simples e apresentou resultados satisfatórios. Já, no caso do AO a partir de textos, os resultados mostram a necessidade de adaptação dos metamodelos das ferramentas para o metamodelo padrão.

A representação inicial aprendida a partir do metadado padrão IMS LD inspirou o interesse pelo estudo do funcionamento do padrão e o quanto a construção de uma ontologia baseada nele pode contribuir para a WSE.

Um sistema educacional utilizado para projetar e executar uma unidade de aprendizagem precisa codificar a semântica do padrão IMS LD na linguagem de programação na qual foi desenvolvida, como Reload, para permitir, além do seu compartilhamento, a interoperabilidade com os demais sistemas educacionais.

Assim, qualquer modificação ocorrida na especificação do padrão gera a necessidade de manutenções em todos os sistemas educacionais envolvidos. Com a existência de uma ontologia para o IMS LD, os conceitos são precisamente definidos e explicitamente descritos, não ocorrendo a necessidade de sua codificação interna nos sistemas. Qualquer alteração na definição do conceito refletirá na ontologia e não nos sistemas de criação ou execução. Um mecanismo de raciocínio associado à linguagem de representação da ontologia pode verificar a consistência da unidade de aprendizagem tanto na fase de criação como na fase de execução. Neste caso representaria uma ontologia de tarefa, de modelagem de conhecimento, de representação ou de aplicação, segundo os tipos de ontologias apresentados no Quadro 1.

Já uma ontologia de domínio construída com base neste padrão permite o reuso e interoperabilidades das unidades de aprendizagem por serviços educacionais inteligentes na WSE e agentes pedagógicos que podem localizar, selecionar, agrupar e integrar unidades de aprendizagem de diferentes servidores educacionais. Estes, por sua vez, podem também divulgar bons projetos instrucionais entre instituições de ensino e permitir aos estudantes encontrar o projeto instrucional mais adequado do curso desejado com suas características e contexto pessoais ou profissionais. Isto tudo independente da tecnologia utilizada para execução do mesmo.

As duas ontologias têm como vantagem secundária a possibilidade do aprendizado de suas instâncias uma vez que muitas unidades de aprendizagem já existem em base de dados ou em arquivos XML dos sistemas educacionais existentes.

Logo, o engenheiro de ontologia pode projetar Ontologias de Aplicação da Unidade de Aprendizagem (OAUA) e uma Ontologia de Domínio do *Learning Design* (ODLD). As primeiras para compatibilizar os metadados existentes da unidade de aprendizagem nos diferentes sistemas educacionais com o padrão IMS LD. A segunda, totalmente aderente à especificação do IMS, participa das Ontologias do Domínio da Educação (ODE₂), e disponibiliza as unidades de aprendizagem para os agentes pedagógicos e serviços educacionais da WSE. A Figura 26 apresenta a arquitetura proposta.

Figura 26 – Arquitetura para Ontologias no Domínio da Educação

O mesmo modelo pode ser aplicado a outras especificações padrão existentes para os recursos educacionais como objetos de aprendizagem e os perfis de estudantes. Finalmente, este trabalho permitiu reunir os complexos conceitos envolvidos com a Web Semântica com os atuais conceitos da Engenharia de Software no domínio da Educação apontando para o uso de padrões como forma de dinamizar a entrega de ontologias.

6.2 Propostas de Trabalhos Futuros

Os resultados obtidos neste trabalho abrem perspectivas para estender o *framework* proposto para realizar as seguintes propostas de trabalhos:

- Realizar o aprendizado de instâncias das ontologias aprendidas, ou seja, associar as representações obtidas da ontologia às instâncias correspondentes nos sistemas existentes presentes principalmente em sistemas gerenciadores de banco de dados e esquemas XML;
- Integrar as representações iniciais obtidas da ontologia a partir de diferentes sistemas existentes visando a construção de uma ontologia de domínio. Unir representações ainda não avaliadas é complexo, ainda mais em se tratando de origens semânticas distintas (formal, para os *schematas* e informal, para os textos);

- Inserir um repositório para os arquivos XML com a aplicação das especificações *Knowledge Discovery Meta-Model* (KDM) e *Abstract Syntax Tree Meta-Model* (ASTM) do OMG para tratamento e correspondência entre as diferentes versões dos padrões utilizados pelo *framework*;
- Realizar o aprendizado de ontologias a partir de outros recursos de saída do sistema existente como demais diagramas UML e ferramentas de desenvolvimento para encontrar outros conceitos e relações existentes no domínio;
- Realizar todas atividades da engenharia progressiva.

Além disso, na WSE propõem-se os seguintes trabalhos:

- Implementar a ontologia de domínio baseada no metadado padrão IMS LD em servidores educacionais;
- Investigar os benefícios e vantagens da construção de ontologias de aplicação nos sistemas educacionais existentes – COL e Teleduc;
- Inserir a ontologia IMS LD no projeto Tidia-Ae.

Certamente ainda podem existir muitas outras propostas para se alcançar o pleno conhecimento dos recursos da Web Semântica e da Engenharia de Ontologias aplicados ao domínio da Educação, pois são áreas desafiadoras e pouco exploradas juntas.

REFERÊNCIAS

ALMEIDA, M. Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção. *Revista Ciência da Informação*, v. 32, n. 3, 2003.

AMORIM, R. R.; LAMA, M.; VILA, S. Using ontologies to model and execute IMS Learning Design documents. In: IEEE INTERNATIONAL CONFERENCE ON ADVANCED LEARNING TECHNOLOGIES, 6., 2006, Kerkrade, The Netherlands. *Proceedings*. IEEE Computer Society, p. 11115-11116, 2006. Disponível em: <<http://doi.ieeecomputersociety.org/10.1109/ICALT.2006.344>>. Acesso em: 10 ago. 2007.

ANTONIOU, G.; VAN HARMELEN, F. Web Ontology Language: OWL. *Handbook on Ontologies in Information Systems*, 2003, p. 67-92. Disponível em: <<http://citeseer.ist.psu.edu/604925.html>>. Acesso em: 10 ago. 2007.

ARAÚJO, M. de *Educação à distância e a Web Semântica: modelagem ontológica de materiais e objetos de aprendizagem para a plataforma COL*. 2003. 178p. Tese (Doutorado) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2003.

ARPÍREZ, J. C.; CORCHO, O.; FERNÁNDEZ-LÓPEZ, M.; GÓMEZ-PÉREZ, A. WebODE: a scalable workbench for ontological engineering. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE, 1., 2001, British Columbia, Canadá. *Proceedings*. New York: ACM Press, 2001. p. 6-13.

ARROYO, L; DOLOG, P; HOUBEN, G; KRAVCIK, M; NILSSON, M; WILD, F. Interoperability in personalized adaptive learning. *Educational Technology & Society Journal*, v. 9, n.2, p. 4-18, 2006.

ASTROVA, I. Reverse engineering of relational databases to ontologies. In: EUROPEAN SEMANTIC WEB SYMPOSIUM, 1., 2004, Heraklion, Greece. *The Semantic Web: Research and Applications*. Heraklion: Springer, 2004. p. 327-341.

ASTROVA, I.; STANTIC, B. An HTML-form-driven approach to reverse engineering of relational databases to ontologies. In: INTERNATIONAL CONFERENCE ON DATABASES AND APPLICATIONS, 2005, Innsbruck, Austria. *Databases and Applications*. Austria: ACTA Press, 2005. p. 246-251.

BACLAWSKI, K.; KOKAR, M.; KOGUT, P.; HART, L.; SMITH, J.; LETKOWSKI, J.; EMERY, P. Extending the Unified Modeling Language for ontology development. *Software and System Modeling*, v. 1, n. 2, p. 142-156, 2002.

BAO, J.; HONAVAR, V. G. Collaborative ontology building with Wiki@nt. In: INTERNATIONAL WORKSHOP ON EVALUATION OF ONTOLOGY BUILDING TOOLS, 3., 2004, Hiroshima, Japan. *Proceedings*. Disponível em: <http://km.aifb.uni-karlsruhe.de/ws/eon2004/EON2004_Proceedings.pdf>. Acesso em: 10 jan. 2005.

BECKETT, D. The design and implementation of the Redland RDF application framework. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, 10., 2001, Hong Kong. *Proceedings*. New York: ACM Press, 2001. p. 449-456.

BELLONI, M. L. *Educação a Distância*. 3. ed. Campinas: Autores Associados, 2003, 108p.

BENJAMINS, V. R.; FENSEL, D.; DECKER, ST.; GÓMEZ-PÉREZ, A. (KA)2: building ontologies for the Internet: a mid-term report. *International Journal of Human-Computer Studies*, v. 51, n. 3, p. 687-712, 1999.

BERGAMASCHI, S; BOUQUET, P; CIACCIA, P; MERIALDO, P. Speaking words of WISDOM: Web intelligent search based on domain ontologies. In: ITALIAN SEMANTIC WEB, 2nd, 2005, Trento, Italia. *Semantic Web Applications and Perspectives 2005*. (CEUR Workshop *Proceedings*, 2005, v. 166). Disponível em: <<http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-166/>>. Acesso em: 10 jan. 2006.

BERGGREN, A.; BURGOS, D.; FONTANA, J.M.; HINKELMAN, D.; HUNG, V.; HURSH A.; TIELEMANS, G. Practical and pedagogical issues for teacher adoption of IMS Learning Design standards in Moodle LMS. *Journal of Interactive Media in Education*, 2005. Disponível em: <<http://www-jime.open.ac.uk/2005/02/>>. Acesso em: 17 ago. 2007.

BERKOWITZ, J. *Network Inference Technology White Paper*. In: UNITED KINGDOM: NETWORK INFERENCE LTD. 2003. Disponível em: <http://www.networkinference.com/downloads/Whitepaper_Technology.pdf>. Acesso em: 21 set. 2006.

BERNARAS, A.; LARESGOITI, I.; CORERA, J. Building and reusing ontologies for electrical network applications. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, 12., 1996, Budapest, Hungary. *Proceedings*. New York: John Wiley, 1999. p. 298-302.

BERNARDES, J.; FERREIRA, M.A.G.; CANTELE, R.C.; TORI, R. Na eletronic computer graphics course: Interlab's Learning action in Tidia-Ae. Technical Report, Internal Document, Tidia-Ae Project, 2008.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The Semantic Web. *Scientific American*, v. 284, n. 5, p. 28-37, 2001.

BIEMANN, C. A Ontology Learning from text: a survey of methods. *Journal LDV Forum*, v. 20, n. 2, p. 75-93, 2005.

BOARETTO, R.; NUNES, C. ; FILATRO, A. Representação de uma ação de aprendizagem através do IMS-Learning Design e implicações para o desenvolvimento de LMSs. In: CONGRESSO NACIONAL DE EDUCAÇÃO A DISTÂNCIA, 8., 2006, Brasília. Anais do VIII Congresso Nacional de Educação a Distância, 2006.

BRESSAN, G.; RUGGIERO, W. Modelo Conceitual e Glossário. Relatório Técnico, Documento Interno, Projeto Tidia-Ae, 2005.

BOUCHIHA, D.; MALKI, M.; BENSLIMANE, S. M. Ontology based web application reverse-engineering approach. *Journal of Computer Science*, v. 6, n. 1, p. 37-46, 2007.

BROCKMANS, S.; HAASE, P.; HITZLER, P.; STUDER, R. A metamodel and UML Profile for rule-extended OWL DL ontologies. In: EUROPEAN SEMANTIC WEB CONFERENCE, 3, 2006, Budva, Montenegro. *The Semantic Web: Research and Applications*. Berlin: Springer, 2006, v. 4011, p. 303-316.

BROEKSTRA, J.; KAMPMAN, A.; HARMELEN, F. V. Sesame: a generic architecture for storing and querying RDF and RDF Schema. In: INTERNATIONAL SEMANTIC WEB CONFERENCE, 1, 2002, Sardinia, Italy. *Proceedings*. Berlin: Springer, 2003. p. 54-68.

BUFFA, M.; DEHORS, S.; FARON-ZUCKER, C.; SANDER, P. Towards a corporate Semantic Web approach in designing learning systems: review of the TRIAL solution project. In: INTERNATIONAL WORKSHOP ON APPLICATIONS OF SEMANTIC WEB TECHNOLOGIES FOR E-LEARNING, 4., 2005, Amsterdam, Netherlands. *Proceedings*. Disponível em: <<http://www.win.tue.nl/SW-EL/2005/swel05-aied05/proceedings/SW-EL05-AIED05-proceedings.pdf>>. Acesso em: 10 mar. 2006.

BUITELAAR, P.; OLEJNIK, D.; SINTEK, M. OntoLT: A Protégé Plug-In for Ontology Extraction from Text. In: EUROPEAN SEMANTIC WEB SYMPOSIUM (ESWS), 1, 2004, Heraklion, Greece. *Proceedings. The Semantic Web: Research and Applications*. Berlin: Springer, 2004, v. 3053, p. 31-44.

BURGOS, D.; TATTERSALL, C.; DOUGIAMAS, M.; VOGTEN, H.; KOPER, R. A first step mapping IMS Learning Design and Moodle. *Journal of Universal Computer Science*, v. 13, n. 7, p. 924-931, 2007.

CAMPOS, F.C.A.; DA ROCHA, A.R.C.; DE CAMPOS, G.H.B. *Design instrucional e construtivismo: em busca de modelos para o desenvolvimento de software*. In: CONGRESSO REDE IBERO-AMERICANA DE INFORMÁTICA EDUCATIVA (RIBIE), IV, Brasília, 1998. Disponível em: <<http://www.niee.ufrgs.br/ribie98/TRABALHOS/250M.PDF>>. Acesso em: 17 ago. 2007.

CARDINAELS, K.; MEIRE, M.; DUVAL, E. Automating metadata generation: the simple indexing interface. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, 14., 2005, Japão. *Proceedings*. New York: ACM, 2005. p. 548-556.

CARR, L.; MILES-BOARD, T.; WOUKEU, A.; WILLS, G.; HALL, W. The case for explicit knowledge in documents. In: ACM SYMPOSIUM ON DOCUMENT ENGINEERING, 2004, Milwaukee, Wisconsin, USA. *Proceedings*. New York: ACM, 2004. p. 90-98.

CARTELLI, A. Semantics, ontologies and information systems in education: concerns and proposals. In: INFORMING SCIENCE + IT EDUCATION CONFERENCE, 2006, Manchester, England. *Proceedings. The Journal of Issues in Informing Science and Information Technology: The Information Universe*, v. 3, p. 113-126, 2006. Disponível em: <http://informingscience.org/Proceedings/InSITE2006/IISIT%20Vol.%203index.htm> . Acesso em: 10 nov. 2007.

CASANOVAS, P.; CASELLAS, N.; TEMPICH, C.; VRANDECIC, D.; BENJAMINS, V. R. OPJK and DILIGENT: ontology modeling in a distributed environment. *Artificial Intelligence and Law*, v. 15, n. 2, p. 171-186, 2007.

CECCARONI, L.; KENDALL, E.F. A graphical environment for ontology development. In: INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS & MULTIAGENT SYSTEMS (AAMAS), 2, 2003, Australia. *Proceedings*. New York: ACM Press, 2003. p. 958-959.

CERBAH, F. Learning Highly Structured Semantic Repositories from Relational Database: The RDBToOnto Tool. ESW2008, Tenerife, Espanha.

CHEN, W; MIZOGUCHI, R. Communication Content Ontology for Learner Model Agent in Multi-agent Architecture. In: WORKSHOP ON ONTOLOGIES FOR INTELLIGENT EDUCATIONAL SYSTEMS, 1999, France. *Proceedings*. Disponível em: <http://www.ei.sanken.osaka-u.ac.jp/aied99/apapers/W-Chen.pdf> . Acesso em: 17 mar. 2007.

CHOI, N.; SONG, I.; HAN, H. A survey on ontology mapping. *SIGMOD Record*, v. 35, n. 3, p. 34-41, 2006.

CIMIANO, P. *Ontology learning and population from text: algorithms, evaluation and applications*. New York: Springer-Verlag, 2006, 171p.

CLARK, K.; PARSIA, B.; HENDLER, J. *Will the Semantic Web change Education?* *Journal of Interactive Media in Education*, 2004. Disponível em: www-jime.open.ac.uk/2004/3 . Acesso em: 17 ago. 2007.

CORCHO O., GÓMEZ-PÉREZ A., GONZÁLEZ-CABERO R., SUÁREZ-FIGUEROA M.C. ODEVAL: a tool for Evaluating RDF(S), DAML+OIL, and OWL concept taxonomies. In: INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE APPLICATIONS AND INNOVATIONS (AIAI), 1, 2004, Toulouse, France. *Proceedings*. Toulouse : Kluwer, 2004. p. 369-382.

CORCHO, O; GÓMEZ-PÉREZ, A. A layered model for building ontology translation systems. *Journal Semantic Web Information System*, v. 1, n. 2, p. 22-48, 2005.

CRANEFIELD, S.; PAN, J. Bridging the gap between the model-driven architecture and ontology engineering. *International Journal of Man-Machine Studies*, v. 65, n. 7, p. 595-609, 2007.

CRISTANI M.; CUEL R. Methodologies for the Semantic Web: state-of-the-art of ontology methodology. *SIGSEMIS Bulletin*, v. 1, n. 2, 2004.

DAGGER, D.; O'CONNOR, A.; LAWLESS, S.; WALSH, E.; WADE, V. P. Service-Oriented E-Learning platforms. *IEEE Computer Society*, v. 11, n. 3, p. 28-35, 2007.

DE HOOG, R.; KABEL, S.; BARNARD, Y.; BOY, G.; DELUCA, P.; DESMOULINS, C.; RIEMERSMA, J.; VERSTEGEN, D. *Re-using technical manuals for instruction: creating instructional material with the tools of the IMAT project*. In: INTELLIGENT TUTORING SYSTEMS (ITS), 2002, Spain. Disponível em: <<http://www.einsteiner.com/suus/Publications/ITS2002b.pdf>>. Acesso em: 10 ago. 2004.

DE NICOLA, A; MISSIKOFF, M; NAVIGLI, R. A proposal for a Unified Process for ontology building: UPON. In: INTERNATIONAL CONFERENCE DATABASE AND EXPERT SYSTEMS APPLICATIONS, 16., 2005, Dinamarca. *Proceedings*. Berlin: Springer, 2005. p. 655-664.

DENNY, M. *Ontology building: a survey of editing tools*. 2002. Disponível em : <<http://www.xml.com/pub/a/2002/11/06/ontologies.html?page=2>>. Acesso em: 15 mar. 2005.

DEVEDŽIĆ, V. *Semantic Web and Education*. Nova York: Springer, 2006, 353p.

DEVEDŽIĆ, V; JOVANOVIĆ, J.; GAŠEVIĆ, D. The pragmatics of current E-Learning standards. *IEEE Internet Computing*, v. 11, n. 3, p. 19-27, 2007.

DICHEVA, D.; SOSNOVSKY, T.G.; BRUSILOVSKY, P. Ontological Web portal for educational ontologies. In: WORKSHOP ON APPLICATIONS OF SEMANTIC WEB TECHNOLOGIES FOR E-LEARNING, 4., 2005, Amsterdam, The Netherlands. *Proceedings*. Disponível em: <<http://www.win.tue.nl/SW-EL/2005/swel05-aied05/Proceedings/4-Dicheva-final-full.pdf>>. Acesso em: 17 ago. 2007.

DJURIĆ, D.; GAŠEVIĆ, D.; DEVEDŽIĆ, V. Ontology modeling and MDA. *Journal of Object Technology*, v. 4, n. 1, p. 109-128, 2005.

DOAN, A.; MADHAVAN, J.; DOMINGOS, P.; HALEVY, A. Y. Learning to map between ontologies on the Semantic Web. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, 11., 2002, Hawaii. *Proceedings*. New York: ACM, 2002. p. 662-673.

DOMINGUE, J. *Tadzebao and Webonto: discussing, browsing and editing ontologies on the Web*. In: KNOWLEDGE ACQUISITION WORKSHOP, 11., 1998, Canada. Disponível em: <<http://ksi.cpsc.ucalgary.ca/KAW/KAW98/domingue/>>. Acesso em: 18 nov. 2004.

DOWNES, s. *The Learning Marketplace: Meaning, Metadata and Content Syndication in the Learning Object Economy.*, 2004. Disponível em: <http://www.downes.ca/files/book3.htm>. Acesso em: 2 Mai. 2008.

DZBOR, M.; MOTTA, E.; DOMINGUE, J. *Opening up magpie via semantic services*. In: INTERNATIONAL SEMANTIC WEB CONFERENCE NOVEMBER, 3rd, 2004, Hiroshima, Japan.

EUZENAT, J. *Corporate memory through cooperative creation of knowledge based and hyper-documents*. In: KNOWLEDGE ACQUISITION WORKSHOPS AND ARCHIVES (KAWA), 10., 1996, Canada. Disponível em: <<http://ksi.cpsc.ucalgary.ca/KAW/KAW96/euzenat/euzenat96b.html>>. Acesso em: 18 nov. 2004.

FALKOVYCH, K.; SABOU, M.; STUCKENSCHMIDT, H. *UML for the Semantic Web: transformation-based approaches*. Knowledge Transformation for the Semantic Web. Amsterdam: IOS Press, 2003. p. 92-106.

FARQUHAR, A.; FIKES, R.; RICE, J. The ontolingua server: a tool for colaborative ontology construction. *International Journal of Human-Computer Studies*, v. 46, n. 6, p. 707-727, 1997.

FAURE, D.; POIBEAU, T. First experiments of using semantic knowledge learned by ASIUM for information extraction task using INTEXT. In *Proceedings of Ontology Learning ECAI-2000 Workshop*, v. 31, p. 7-12, 2000.

FENSEL, D. *OIL and DAML + OIL: Ontology Languages for the Semantic Web*. 2003. Disponível em: <<http://dx.doi.org/10.1002/0470858060.ch2>>. Acesso em: 12 ago. 2005.

FENSEL, D.; BUSSLER, C. The Web Service Modeling Framework WSMF. *Electronic Commerce Research and Applications*, v. 1, n. 2, p. 113-137, 2002.

FERNANDEZ-LOPEZ, M. Overview of methodologies for building ontologies. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE (IJCAI), 16., 1999, Suécia. *Proceedings*. (CEUR Workshop Proceedings, 1999, v. 18, n. 4.) Disponível em: < <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-18/>>. Acesso em: 10 mar. 2003

FERREIRA, M. A. G. F.; TORI, R. Da representação gráfica à realidade virtual: experiências bem sucedidas, novas propostas. In: CONGRESSO BRASILEIRO DE ENSINO DE ENGENHARIA, 26., 1998, São Paulo. Anais. São Paulo: ABENGE, 1998, p. 1613-1626.

FREITAS, F. *Ontologias e a Web Semântica*. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, XXIII, 2003, Campinas, Brasil. Anais. Campinas: SBC, 2003, v. 8. p. 1-52.

GANGEMI, A.; GÓMEZ-PÉREZ, A.; PRESUTTI, V.; SUAREZ-FIGUEROA M. C. Knowledge Reuse and Reengineering over the Semantic Web. In: EUROPEAN SEMANTIC WEB CONFERENCE (ESWC), 5., 2008, Tenerife, Espanha. *Proceedings*.

GAŠEVIĆ, D.; DJURIĆ, D.; DEVEDŽIĆ, V. *Model Driven Architecture and ontology development*. Berlin: Springer, 2006, 311p.

GAŠEVIĆ, D.; DJURIĆ, D.; DEVEDŽIĆ, V.; DAMJANOVI, V. Converting UML to OWL ontologies. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, 13., 2004, New York. *Proceedings*. New York: ACM, 2004. p. 488-489.

GAŠEVIĆ, D.; JOVANOVIĆ, J.; DEVEDŽIĆ, V.; BOŠKOVIĆ, M. Ontologies for reusing learning object content. In: INTERNATIONAL WORKSHOP ON APPLICATIONS OF SEMANTIC WEB TECHNOLOGIES FOR E-LEARNING, 4., 2005, Amsterdam, The Netherlands. *Proceedings*. Disponível em: <<http://www.win.tue.nl/SW-EL/2005/swel05-icalt05/final/W3-1.pdf>>. Acesso em: 10 nov. 2007.

GÓMEZ-PÉREZ, A. Ontology engineering methodologies. In: INTERNATIONAL SEMANTIC WEB CONFERENCE, 6., 2007, Busan, Korea. *Proceedings*. Disponível em: <<http://kmi.open.ac.uk/events/iswc07-semantic-web-intro/pdf/4.%20Ontology%20Engineering%20Methodologies.pdf>>. Acesso em: 05 nov. 2007.

GÓMEZ-PÉREZ, A.; CORCHO, O. Ontology languages for the Semantic Web. *Intelligent Systems*, v.17, n.1, p. 54-60, 2002.

GÓMEZ-PÉREZ, A.; MANZANO-MACHO, D. *A survey of ontology learning methods and techniques*. Projeto OntoWeb, Deliverable 1.5, 2003. Disponível em: <<http://www.sti-innsbruck.at/fileadmin/documents/deliverables/Ontoweb/D1.5.pdf>>. Acesso em: 10 out. 2003.

GOTH, G. Will the Semantic Web quietly revolutionize Software Engineering? *IEEE Software*, v. 24, p. 100-105, 2007.

GRUBER, T. R. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, v. 43, p. 907-928, 1995.

GUARINO, N. Understanding, building, and using ontologies: a commentary to using explicit ontologies in KBS development. *International Journal of Human and Computer Studies*, v. 46, p. 293-310, 1997.

HAASE, P.; VOLKER, J. Ontology learning and reasoning - dealing with uncertainty and inconsistency. In: INTERNATIONAL SEMANTIC WEB CONFERENCE (ISWC), 4., 2005, Ireland. *Proceedings*. (CEUR Workshop Proceedings, 2005, v. 173.) Disponível em: <<http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-173/>>. Acesso em: 10 ago. 2004.

HAAV, H.; LUBI, T. *A survey of concept-based information retrieval tools on the Web*. In: ADVANCES IN DATABASES AND INFORMATION SYSTEMS, 5., 2001, Lithuania. *Proceedings*. Berlin: Springer, 2001. p. 29-41. Disponível em: <<http://www.science.mii.lt/ADBIS/local2/haav.pdf>>. Acesso em: 05 set. 2004.

HANDSCHUH, S.; STAAB, S. CREAM: CREATing Metadata for the Semantic Web. *Computer Networks*, v. 42, n. 5, p. 579-598, 2003.

HATALA, M.; RICHARDS, G.; EAP, T.; WILLMS, J. The interoperability of learning object repositories and services: standards, implementations and lessons learned. In: WORLD WIDE WEB CONFERENCE, 13., 2004, New York, USA. *Proceedings*. New York: ACM, 2004, p. 19-27.

HEW, K. F. Past Technologies, Practice and Applications: A Discussion on How the Major Developments in Instructional Technology in the 20th Century Affect the Following Qualities ? Access, Efficiency, Effectiveness, and Humaneness. In: CONFERENCE OF THE ASSOCIATION FOR EDUCATIONAL COMMUNICATIONS AND TECHNOLOGY, 27, 2004, Chicago, AETC.

HOGUE, A.; KARGER, D. *Thresher: automating the unwrapping of semantic content from the world wide web*, In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, 14., 2005, Chiba, Japan. Disponível em: < <http://www2005.org/cdrom/docs/p86.pdf>>. Acesso em: 10 nov. 2006.

HOWELL, S. L.; WILLIAMS, P. B.; LINDSAY, N. K. Thirty-two trends affecting distance education: an informed foundation for strategic planning. *Online Journal of Distance Learning Administration*, v. VI, n. III, 2003. Disponível em: <<http://www.westga.edu/~distance/ojdla/fall63/howell63.html>>. Acesso em: 10 jan. 2008.

HUMMEL, H. G. K.; MANDERVELD, J. M.; TATTERSALL, C.; KOPER, E. J. R. Educational Modelling Language: new challenges for instructional re-usability and personalized learning. *International Journal of Learning Technology*, v. 1, n. 1, p. 110-111, 2004.

IHSAN, I.; REHMAN, M.; AHMED, M.; QADIR, M.A. UREKA Learning-Object Taxonomy & Repository Architecture – ULTRA. In: IEEE INTERNATIONAL MULTITOPIC CONFERENCE (INMIC), 10., 2006, Islamabad, Pakistan. *Proceedings*. IEEE Computer Society Press, p. 231–236, 2006.

IMS Global Learning Consortium. *Learning Design Specification*. 2007. Disponível em: <<http://www.imsglobal.org/learningdesign/>>. Acesso em: 10 jan. 2008.

INTERLAB. Engine: a didactic game engine in java. 2008. Disponível em: <<http://www.engine.org/>>. Acesso em: 10 jun 2008.

ISO: *International Organization for Standardization*. *ISO/IEC 13250:2000 Topic Maps: Information Technology -- Document Description and Markup Languages*. 1999. Disponível em: <<http://www.y12.doe.gov/sgml/sc34/document/0129.pdf>>. Acesso em: 10 jan. 2008.

JASPER, R.; USCHOLD, M. A framework for understanding and classifying ontology applications. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE (IJCAI), 16., 1999, Stockholm, Sweden. *Proceedings*. (CEUR Workshop Proceedings, 1999, v. 18, n. 11.) Disponível em: < <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-18/>>. Acesso em: 10 abr. 2003.

JIMÉNEZ-RUIZ, E; BERLANGA, R. A view-based methodology for collaborative ontology engineering: an approach for complex applications (VIMethCOE). In: IEEE INTERNATIONAL WORKSHOPS ON ENABLING TECHNOLOGIES: INFRASTRUCTURES FOR COLLABORATIVE ENTERPRISES (WETICE), 15., 2006, Manchester, England. *Proceedings*. IEEE Computer Press, 2006. p. 376-381.

JOVANOVIĆ, J.; GAŠEVIĆ, D.; KNIGHT, C.; RICHARDS, G. Ontologies for effective use of context in e-Learning settings. *Journal of Educational Technology & Society*, v. 10, n. 3, p. 47-59, 2007.

KASKALIS, T. H.; TZIDAMIS, T. D.; MARGARITIS, K. Multimedia Authoring Tools: The Quest for an Educational Package. *Educational Technology & Society*, v. 10, n. 3, p. 135-162, 2007.

KEENOY, K.; LEVENE, M.; PETERSON, D. *SeLeNe – Self E-Learning Networks. WP4 Deliverable D4.2: Personalisation and Trails in Self e-Learning Networks*. Technical Report D4.2, Project Selene, 2004. Disponível em: <<http://www.dcs.bbk.ac.uk/selene/reports/Del4.2-2.1.pdf>>. Acesso em: 10 jan. 2008.

KIM, J. A.; CHOI, S. Y. Evaluation of ontology development methodology with CMM-I. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING RESEARCH, MANAGEMENT AND APPLICATIONS, 5., 2007. *Proceedings*. IEEE Computer Society Press, 2007. p. 823-827.

KLJUN, M.; VIČIČ, J.; KAVŠEK, B.; KAVČIČ, A. Evaluating comparisons and evaluations of Learning Management Systems. In: INTERNATIONAL CONFERENCE INFORMATION TECHNOLOGY INTERFACES, 29., 2007, Croatia. *Proceedings*. IEEE Computer Society Press, 2007. p. 363-368.

KOOHANG, A.; DU PLESSIS, J.; BRITZ, J. Learning objects in learning organizations: a knowledge management perspective. *International Journal of Management and Decision Making*, v.7, n. 1, p. 58-70, 2006.

KOPER, R. *Modeling units of study from a pedagogical perspective: the pedagogical meta-model behind EML*. 2001. Disponível em: <<http://dspace.ou.nl/bitstream/1820/36/1/Pedagogical+metamodel+behind+EMLv2.pdf>>. Acesso em: 10 mai. 2004.

KOPER, R.; RODRÍGUES-ARTACHO, M.; LEFRERE, P.; RAWLINGS, A.; ROSMALEN, P. *Survey of Educational Modelling Languages (EMLs)*. 2002. Disponível em: <<http://dspace.ou.nl/bitstream/1820/227/2/eml-report-cen-issr.pdf>>. Acesso em: 10 mai. 2004.

KOPER, R. Use of the Semantic Web to solve some basic problems in Education: increase flexible, distributed lifelong learning, decrease teachers' workload. *Journal of Interactive Media in Education, Special Issue on the Educational Semantic Web*, 2004. Disponível em: <<http://www-jime.open.ac.uk/2004/6/>>. Acesso em: 17 ago. 2007.

KOTIS, K.; VOUIROS, G. Human-centered ontology engineering: the HCOME methodology. *International Journal of Knowledge and Information Systems*, v. 10, n. 1, p. 109-131, 2006.

LANFRANCHI, V.; CIRAVEGNA, F.; PETRELLI, D. *Semantic Web-based document: editing and browsing in AktiveDoc*, In: EUROPEAN SEMANTIC WEB CONFERENCE, 2nd, 2005, Heraklion, Greece. Disponível em: <<http://www.dcs.shef.ac.uk/~fabio/paperi/AktiveDocESWC04.pdf>>. Acesso em: 10 jun. 2006.

LARA, M. L. G. Diferenças conceituais sobre termos e definições e implicações na organização da linguagem documentária. *Ciência da Informação*, v. 333, p. 91-96, 2004.

LARMAN, C. *Utilizando UML e padrões*. Porto Alegre: Bookman, 2007, 695p.

LEGER, A.; MICHEL, G.; BARRETT, P.; GITTON, S.; GOMEZ-PEREZ, A.; LEHTOLA, A.; MOKKILA, K.; RODRIGEZ, S.; SALLANTIN, J.; VARVARIGOU, T.; VINESSE, J. *Ontology domain modeling support for multi-lingual services in E-Commerce: MKBEEM*. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, 14., 2000, Germany. Disponível em: <<http://mkbeem.elibel.tm.fr/paper/ecai00-final.pdf>>. Acesso em: 22 dez. 2001.

LENAT, D. B.; GUHA, R. V. *Building large knowledge-based systems; representation and Inference in the Cyc Project*. Massachusetts: Addison-Wesley, 1990, 372p.

LI, M.; DU, X.; WANG, S. Learning ontology from relational database. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND CYBERNETICS, 2005, GuangZhou, China. *Proceedings*. IEEE Press, 2005, v. 6. p.3410-3415. Disponível em: < [http://iir.ruc.edu.cn/nsf/2005/icmlc05\(liman\).pdf](http://iir.ruc.edu.cn/nsf/2005/icmlc05(liman).pdf)>. Acesso em: 10 ago. 2006.

LOZANO-TELLO, A.; GOMEZ-PEREZ, A. ONTOMETRIC: a method to choose the appropriate ontology. *Journal of Database Management*, v. 15, n. 2, 2004.

MACLEOD, D. *Learning object repositories: deployment and diffusion*. CANARIE E-Learning Program, 2005. Disponível em: <http://www.canarie.ca/funding/elearning/2005_LOR_report.pdf>. Acesso em: 10 ago. 2007.

MAEDCHE, A. *Ontology learning for the Semantic Web*. Massachusetts: Kluwer Academic Publishers, 2002, 241p.

MALAXA, V.; DOUGLAS, I. A framework for metadata creation tools. *Interdisciplinary Journal of Knowledge and Learning Objects*, v. 1, p. 151-162, 2005.

MARIETTO, M. G. B.; DAVID, N.; SICHMAN, J. S.; COELHO, H. *Infraestrutura para a Construção de Ontologias*. Technical Report n. 01-2002, Escola Politécnica da Universidade de São Paulo, 2002.

MCBRIDE, B. Jena: implementing the RDF model and syntax specification. In: INTERNATIONAL WORKSHOP ON THE SEMANTIC WEB, 2., 2001. *Proceedings*.

(CEUR Workshop *Proceedings*, v. 40, 2001.) Disponível em: <<http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-40/>>. Acesso em: 14 mai. 2004.

MERRILL, M. D. First principles of instruction. *Educational Technology Research and Development*, v. 50, n. 3, p. 43-59, 2002.

MILLER, L. *Inkling architectural overview*, 2001. Disponível em: <<http://ilrt.org/discovery/2001/07/inkling/>>. Acesso em: 10 mar. 2005.

MILSTEAD, J.; FELDMAN, S. Metadata: cataloging by any other name. *Magazine Online*, v. 23, n. 1, p. 25-31, 1999.

MIZOGUCHI, R.; HAYASHI, Y.; BOURDEAU, J. Inside theory-aware authoring system. In: SWEL WORKSHOP OF ONTOLOGIES AND SEMANTIC WEB SERVICES FOR IES, 5., 2007, USA. *Proceedings*. Disponível em: <<http://www.ei.sanken.osaka-u.ac.jp/pub/miz/SWEL07-miz.pdf>>. Acesso em: 10 nov. 2007.

MIZOGUCHI, R.; TIJERINO, Y.; IKEDA, M. Task analysis interview based on task ontology. *Expert Systems With Applications*, v. 9, n. 1, p. 15-25, 1995.

NAJJAR, J.; KLERKX, J.; VUORIKARI, R.; DUVAL, E. Finding appropriate learning objects: an empirical evaluation. In: EUROPEAN CONFERENCE ON RESEARCH AND ADVANCED TECHNOLOGY FOR DIGITAL LIBRARIES, 9., 2005, Austria. *Proceedings*. Berlin: Springer, 2005, v. 3652. p. 323-335.

NAKAMURA, R.; BERNARDES, J.; TORI, R. enJine: architecture and application of an open-source didactic game engine. In: BRAZILIAN SYMPOSIUM ON COMPUTER GAMES AND DIGITAL ENTERTAINMENT, 5., 2006, Recife. Disponível em: <http://www.cin.ufpe.br/~sbgames/proceedings/files/enJine.pdf>. Acesso em: 10 mar 2008.

NAVIGLI, R., VELARDI, P. Learning domain ontologies from document warehouses and dedicated web sites. *Journal Computational Linguistics*, v. 30, n. 2, p. 151-179, 2004.

NAVIGLI, R., VELARDI, P. Ontology Enrichment Through Automatic Semantic Annotation of On-line Glossaries. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE ENGINEERING AND KNOWLEDGE MANAGEMENT (EKAW), 15, 2006, Podebrady, Czech Republic. *Proceedings*. Berlin: Springer, 2006, v. 4248, p. 126-140.

NILSSON, M. Harmonization of metadata standards. Technical Report D4.7, PROLEARN, *European Commission Sixth Framework Project (IST-507310)*, 2008. Disponível em: <<http://www.ieeeltsc.org/NewsItems/harmonization-of-metadata-standards>>. Acesso em: 10 mai. 2008.

NOY, N. F.; FERGERSON, R. W.; MUSEN, M. A. The knowledge model of Protégé-2000: combining interoperability and flexibility. In: INTERNATIONAL CONFERENCE

ON KNOWLEDGE ENGINEERING AND KNOWLEDGE MANAGEMENT (EKAW), 2., 2000, Juan-les-Pins, France. *Proceedings*. Berlin: Springer, 2000. p. 69-82.

OBERLE, D.; LAMPARTER, S.; EBERHART, A.; STAAB, S. *Semantic management of web services*. Technical report, University of Karlsruhe, 2005. Disponível em: <<http://www.aifb.uni-karlsruhe.de/WBS/dob/icws2005.pdf>>. Acesso em: 10 mar. 2005.

OMG: *Object Management Group. Ontology Definition Metamodel*. 2007a. Disponível em: <<http://www.omg.org/docs/ptc/07-09-09.pdf>>. Acesso em: 10 jan. 2008.

OMG: *Object Management Group. Model Driven Architecture*. 2007b. Disponível em: <<http://www.omg.org/mda/>>. Acesso em: 10 mar. 2007.

OMG: *Object Management Group. Architecture Driven Modernization*. 2008. Disponível em: <<http://adm.omg.org>>. Acesso em: 10 fev. 2008.

PAQUETTE, G. An ontology and a software framework for competency modeling and management. *Educational Technology & Society*, v. 10, n. 3, p. 1-21, 2007.

PINTO, H.; TEMPICH, C.; SURE, Y. *DILIGENT: Towards a fine-grained methodology for Distributed, Loosely-controlled and evolving Engineering of oNTologies*. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, 16., 2004, Valencia, Spain. Disponível em: <citeseer.ist.psu.edu/article/pinto04diligent.html>. Acesso em: 10 mar. 2005.

PROTÉGÉ. Overview of the Protege Metamodel. 2006. Disponível em: <<http://protege.cim3.net/cgi-bin/wiki.pl?ProtegeXMIMetamodel>>.. Acesso em: 10 nov 2007.

PULIDO, J.R.G.; RUIZ, M.A.G.; HERRERA, R.; CABELLO, E.; LEGRAND, S.; ELLIMAN, D. Ontology languages for the semantic web: a never completely updated review. *Knowledge-Based Systems*, v. 19, n. 7, p. 489-497, 2006.

ROSSANO, V.; JOY, M.; ROSELLI, T.; SUTINEN, E. A taxonomy for definitions and applications of LOs: a metaanalysis of ICALT papers. *Educational Technology & Society Journal*, v. 8, n. 4, p. 148-160, 2005.

SCHREIBER, G.; CRUBEZY, M.; MUSEN, M. A case study in using Protege-2000 as a CASE tool for CommonKADS. In: INTERNATIONAL CONFERENCE KNOWLEDGE ENGINEERING AND KNOWLEDGE MANAGEMENT, 12., 2000, Berlin. *Proceedings*. Berlin: Springer, 2000, v. 1937. p. 33-48.

SCHIMT, D. C. Model Driven engineering. *IEEE Computer Society*, v. 39, n. 2, p. 25-31, 2006.

SHADBOLT, N.; HALL, W. The Semantic Web revisited. *IEEE Intelligent Systems*, v. 21, n. 3, p. 96-101, 2006.

SHADBOLT, N. Garlik: Semantic Technology for the Consumer. In: EUROPEAN SEMANTIC WEB CONFERENCE (ESWC 2008), 5, 2008, Tenerife, Espanha. *Proceedings*. Berlin: Springer, 2008, p. 4.

SHAMSFARD, M.; BARFOROUSH, A. A. Learning ontologies from natural language texts. *International Journal of Human-Computer Studies*, v. 60, n. 1, p. 17-63, 2004.

SHETH, A.; RAMAKRISHNAN, C.; THOMAS, C. Semantics for the Semantic Web: the implicit, the formal and the powerful. *International Journal on Semantic Web and Information*, v. 1, n. 1, p. 1-18, 2005.

SHUM, S. B.; MOTTA, E.; DOMINGUE, J. ScholOnto: Na Ontology-Based Digital Library Server for Research Documents and Discourse. *International Journal on Digital Libraries*, v. 3, n. 3, p. 237-248, 2000.

SIMPERL, E. P. B.; TEMPICH, C. Ontology Engineering: a reality check. In: INTERNATIONAL CONFERENCE ON ONTOLOGIES, DATABASES, AND APPLICATIONS OF SEMANTICS, 5., 2006, Montpellier, France. *Proceedings*. Berlin: Springer, 2006, v. 4275. p. 836-854.

SPYNS, P.; MEERSMAN, R.; JARRAR, M. Data modelling versus ontology engineering. *SIGMOD Record*, v. 31, n. 4, p. 12-17, 2002.

STAAB, S.; MAEDCHE, A. Ontology engineering beyond the modeling of concepts and relations. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE (ECAI), 2000, Berlin. *Proceedings*. Amsterdam: IOS Press, 2000.

STAAB, S.; STUDER, R.; SCHNURR, H.; SURE, Y. Knowledge Processes and Ontologies. *IEEE Intelligent Systems*, v. 16, n. 1, p. 26-34, 2001.

STOJANOVIC, L.; STAAB, S.; STUDER, R. eLearning based on the Semantic Web. In: WORLD CONFERENCE ON THE WWW AND INTERNET (WEBNET), 2001, Orlando, USA. Disponível em: <<http://citeseer.ist.psu.edu/501440.html>>. Acesso em: 10 Maio. 2004.

STUDER, R.; BENJAMINS, V. R.; FENSEL, D. Knowledge Engineering: principles and method. *Data Knowledge Engineering*, v. 25, n. 1-2, p. 161-197, 1998.

SUGUMARAN, V.; STOREY, V. C. The role of domain ontologies in database design: an ontology management and conceptual modeling environment. *ACM Transactions on Database Systems*, v. 31, n. 3, p. 1064-1094, 2006.

SWARTOUT, W.; GIL, Y. *EXPECT: A user-centered environment for the development and adaptation of knowledge-based planning aids*. Advanced Planning Technology: Technological Achievements of the ARPA/Rome Laboratory Planning Initiative. Menlo Park: AAAI Press / MIT Press, 1996. p. 250-258. Disponível em: <<http://citeseer.ist.psu.edu/swartout96expect.html>>. Acesso em: 10 maio 2004.

TALLIS, M. SemanticWord processing for content authors. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE CAPTURE, 2., 2003, Sanibel, Florida, USA, 2003.

TAO, F.; MILLARD, D.; ZALFAN, M.; CHEN, L.; DAVIS, H. *Semantic Web assisted learning experience management – architecture and strategy for collaborative learning experience sharing*. Technical Report, 2006. Disponível em: <<http://eprints.ecs.soton.ac.uk/13024/01/wi2006-TR.pdf>>. Acesso em: 17 ago. 2007.

TATTERSAL, C.; KOPER, R. *EML and IMS Learning Design: from LO to LA*. 2003. Disponível em: < <http://dspace.learningnetworks.org/handle/1820/107>>. Acesso em: 10 jul. 2007.

TEMPICH, C.; PINTO, H. S.; SURE, Y.; VRANDEČIĆ, CASELLAS, N.; CASANOVAS, P. Evaluating Diligent ontology Engineering in a Legal Case Study. In: XXII WORLD CONGRESS OF PHILOSOPHY OF LAW AND SOCIAL PHILOSOPHY, 22., 2005, Granada, Spain. *Proceedings*. Granada: University of Granada, 2005. p. 330-331.

TORI, R.; FERREIRA, M. A. G. F. Aprendizagem em computação gráfica: uma abordagem top-down. In: VI WEI - Workshop sobre Educação em Informática, 1998, Belo Horizonte. Anais do XVIII CSBC - SEMISH - CTD - CTIC - WEI - WIE. Belo Horizonte, MG, Brasil, 1998. v. I. p. 521-531.

TORI, R.; NAKAMURA, R.; BERNARDES, J. Teaching introductory computer graphics using java 3D, games and customized software: a Brazilian experience. In: INTERNATIONAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES, 2006, Boston, Massachusetts. *Proceedings*. New York: ACM SIGGRAPH 2006 Educators program, 2006, p. 12.

TRINKUNAS, J.; VASILECAS, O. A graph oriented model for ontology transformation into conceptual data model. *Information Technology and Control*, v. 36A, n1, 2007. Disponível em: < <http://itc.ktu.lt/itc361/Trinkun361.pdf>>. Acesso em: 10 set. 2007.

ULRICH, H.; PINKWART, N.; OELINGER, M.; ZEINI, S.; VERDEJO, F.; BARROS, B.; MAYORGA, J. I. Building bridges within learning communities through ontologies and “thematic objects”. In: CONFERENCE ON COMPUTER SUPPORT FOR COLLABORATIVE LEARNING (CSCL), 2005, Taipei, Taiwan. *Proceedings*. International Society of the Learning Sciences, 2005. p. 211-220.

UPADHYAYA, S. R.; KUMAR, P. S. ERONTO: a tool for extracting ontologies from extended E/R diagrams. In: SYMPOSIUM ON APPLIED COMPUTING (SAC), 2005, Santa Fe, New Mexico. *Proceedings*. New York: ACM, 2005. p. 666-671.

UREN, V.; CIMIANO, P.; IRIA, J.; HANDSCHUDH, S.; VARGAS-VERA, M.; MOTTA, E.; CIRAVEGNA, F. Semantic annotation for knowledge management: requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web*, v. 4, n. 1, p. 14-28, 2006.

USCHOLD, M.; GRUNINGER, M. Ontologies: principles, methods and applications. *Knowledge Engineering Review*, v. 11, n. 2, 1996.

VERBET, K.; DUVAL E. Towards a global component architecture for Learning objects: a comparative analysis of learning objects content models. In: WORLD CONFERENCE ON EDUCATIONAL MULTIMEDIA, HYPERMEDIA AND TELECOMMUNICATIONS, 2004, Lugano, Suíça. *Proceedings*. Berlin: Springer/Heidelberg, 2005. p. 111-121.

VÖLKER, J.; VRANDECIC, D.; SURE, Y. *Automatic Evaluation of Ontologies (AEON)*. In: INTERNATIONAL SEMANTIC WEB CONFERENCE (ISWC), 4., 2005, Galway, Ireland. *Proceedings*. Berlin: Springer Verlag, v. 3729, 2005. p. 716-731.

W3C: World Wide Web Consortium. *Semantic Web*. 2008. Disponível em: <<http://www.w3.org/2001/sw/>>. Acesso em: 10 mar. 2008.

W3C: World Wide Web Consortium. *OWL Web Ontology Language overview*. 2007. Disponível em: <http://www.w3.org/2007/OWL/wiki/OWL_Working_Group>. Acesso em: 10 dez. 2007.

W3C: World Wide Web Consortium. *Resource Description Framework*. 2004. Disponível em: <<http://www.w3.org/rdf/>>. Acesso em: 10 set. 2006.

WANG, T. H. *A survey of SCORM on authoring*. 2006. Disponível em: <<http://adlcommunity.net/file.php/1/Resources/MINEAuthoring2006-8.pdf>> Acesso em: 20 set. 2007.

WARREN, P. Knowledge management and the Semantic Web: from scenario to technology. *IEEE Intelligent Systems*, v. 21, n. 1, p. 53-59, 2006.

WELTY, C.; KALRA R.; CHU-CARROLL, J. Evaluating ontological analysis. In: INTERNATIONAL SEMANTIC WEB CONFERENCE, 2., 2003, Sanibel Island, USA. *Proceedings*. San Jose: AAAI Press, 2004. p.311-314.

WILEY, D. Connecting learning objects to instructional design theory: a definition, a metaphor, a taxonomy. 2001. Disponível em: <<http://www.reusability.org/read/chapters/wiley.doc>>. Acesso em: 10 mar. 2007.

YOUN, S.; MCLEOD, D. Ontology development tools for ontology-based knowledge management. *Encyclopedia of E-Commerce, E-Government and Mobile Commerce*. London: Idea Group Inc., 2006. p. 858-864. Disponível em: <<http://www-scf.usc.edu/~syoun/>>. Acesso em: 10 set. 2007.

ZAVITSANOS, E.; VOUIROS, G. A.; PALIOURAS, G. *Ontology learning and evaluation: a survey technical report*. Technical Report, Project OntoSum, 2006. Disponível em: <<http://www.ontosum.org/system/files?file=StateOfTheArt.pdf>> Acesso em: 10 out. 2007.

ZDRAHAL, Z.; MULHOLLAND, P.; DOMINGUE, J.; HATALA, M. Sharing engineering design knowledge in a distributed environment. *Behaviour & Information Technology*, v. 19, n. 3, p. 189-200, 2000.

ZOUAQ, A; NKAMBOU, R.; FRASSON, C. An integrated approach for automatic aggregation of learning knowledge objects. *Interdisciplinary Journal of Knowledge and Learning Objects (IJKLO)*, v.3, p. 135-162, 2007.

Referências Complementares

BORGIO, S.; GUARINO, N.; MASOLO, C.; VETERE, G. Using a large linguistic ontology for Internet-based retrieval of object-oriented components. In: CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, 9., 1997, Madrid, Espanha. *Proceedings*. Illinois: Knowledge Systems Institute, 1997. p. 528-534.

BRUNNER, J.; MA, L.; WANG, C.; ZHANG, L.; WOLFSON, D. C.; PAN, Y.; SRINIVAS, K. Explorations in the use of semantic Web technologies for product information management. In: WORLD WIDE WEB CONFERENCE, 16, 2007, Banff, Canada. *Proceedings*. New York: ACM, 2007. p. 747-756.

BUSHMANN, F.; MEUNIER, R.; ROHNERT, H.; SOMMERLAD, P.; STAL, M. *Pattern-Oriented software architecture: a system of patterns*. Chichester: John Wiley & Sons, 1996.

CHEN, H.; FININ, T.; JOSHI, A. Using OWL in a pervasive computing broker. In: INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AND MULTI-AGENT SYSTEMS, WORKSHOP ON ONTOLOGIES IN AGENT SYSTEMS, 2003, Melbourne, Australia. *Proceedings*. (CEUR Workshop Proceedings, 2003, v. 73) Disponível em: < <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-73>>. Acesso em: 10 jun. 2004.

CIMIANO, P.; VÖLKER, J. Text2Onto for ontology learning and data-driven change discovery. In: INTERNATIONAL CONFERENCE ON APPLICATIONS OF NATURAL LANGUAGE TO INFORMATION SYSTEMS, 10, 2005, Alicante, Spain. *Proceedings*. Berlin: Springer, 2005. p. 227-238.

DECKER, S.; MELNIK, S.; VAN HARMELEN, FENSEL, D.; KLEIN, M.; F.; BROEKSTRA, J.; ERDMANN, M.; HORROCKS, I. The Semantic Web: roles of XML and RDF. *IEEE Internet Computing*, v.4, n.5, p. 63-73, 2000.

FAYAD, M. C.; SCHMIDT, D. C.; JOHNSON, R. E. *Building application frameworks – Object-Oriented foundations of frameworks design*. Nova York: J. Wiley, 1999.

FIKES, R; HAYES, P; HORROCKS, I. *OWL-QL – a language for deductive query answering on the Semantic Web*. Stanford University, Technical Report, n. KSL 03-14, 2003.

FREITAS, F.L.G. *Sistemas multiagentes cognitivos para recuperação, classificação e extração integradas de informação da web*. 2002. 226p. Tese (Doutorado) – Universidade Federal de Santa Catarina, Florianópolis, Santa Catarina, 2002.

GÓMEZ-PÉREZ, A.; BENJAMINS, V. R. Applications of ontologies and problem-solving methods. *Artificial Intelligence Magazine*, v. 20, n. 1, p. 119-122, 1999.

HANDSCHUH, S.; MAEDCHE, A.; STOJANOVIC, L.; VOLZ, R. *KAON – the Karlsruhe ontology and Semantic Web infrastructure*. Technical report, Projeto KAON, 2001. Disponível em: <<http://kaon.semanticWeb.org/kaon/white-paper.pdf>>. Acesso em: 10 jan. 2008.

KIFER, M.; LAUSEN, G.; WU, J. Logical foundations of object-oriented and frame-based languages. *Journal ACM*, v. 42, n. 4, p. 741–843, 1995.

KNUBLAUCH, H.; MUSEN, M. A.; RECTOR, A. L. Editing Description Logic Ontologies with the Protégé OWL Plugin. In: INTERNATIONAL WORKSHOP ON DESCRIPTION LOGICS, 2004, Canada. *Proceedings*. (CEUR Workshop Proceedings, v. 104, 2004.) Disponível em: <<http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-104/>>. Acesso em: 20 mar. 2005.

MOTIK, B.; OBERLE, D.; STAAB, S.; STUDER, R.; VOLZ, R. *KAON — Server Architecture*. Technical Report 421, Institute AIFB, University of Karlsruhe, 2002. Disponível em: <<http://wonderweb.man.ac.uk/deliverables/documents/D5.pdf>>. Acesso em: 10 mar. 2005.

MUELLER, H. A.; JAHNKE, J. H.; SMITH, D. B.; STOREY, M. D.; TILLEY, S. R.; WONG, K. Reverse engineering: a roadmap. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, (ICSE), 22., 2000, Ireland. *Proceedings*. New York: ACM, 2000. p. 47-60.

NAVATHE, S. B.; CERI, S.; BATINI, C. *Conceptual database design: an entity-relationship approach*. Redwood: The Benjamin/Cummings Publishing Company, 1992, 455p.

NYULAS, C., O'CONNOR, M., TU, S. Datamaster - a plug-in for importing schemas and data from relational databases into Protégé. In: PROTÉGÉ CONFERENCE, 10, Budapeste, Hungria, 2007. Disponível em: http://protege.stanford.edu/conference/2007/presentations/10.01_Nyulas.pdf>. Acesso em: 10 mar. 2008.

O'BRIEN, P.; ZHAN, C. Domain ontology management environment. In: INTERNATIONAL CONFERENCE ON SYSTEMS SCIENCES (HICSS), 33., 2000, Hawaii. *Proceedings*. IEEE Computer Sciences, v. I, p. 9, 2000. Disponível em: <<http://dlib.computer.org/conference/hicss/0493/pdf/04938015.pdf>>. Acesso em: 10 mar. 2005.

OLIVEIRA, A. A. *Transformações inter-metamodelos: metodologia e padrões*. 2005. 157p. Tese (Doutorado) – Escola Politécnica da Universidade Federal de São Paulo, São Paulo, 2002.

RUSSEL, S.; NORVIG, P. *Inteligência Artificial*. Rio de Janeiro: Elsevier, 2004, 1021p.

SAMPSON, D.G.; LYTRAS, M.D.; WAGNER, G.; DIAZ, P. Ontologies and the Semantic Web for E-learning. *Educational Technology & Society Journal*, v.7, n. 4, p. 26-28, 2004.

SCHREIBER, A.; WIELINGA, B.; AKKERMANS, H.; VAN DE VELDE, W.; ANJEWIERDEN, A. CML: The CommonKADS conceptual modeling language. In: EUROPEAN KNOWLEDGE ACQUISITION WORKSHOP, 8., 1994. *Proceedings*. Berlin: Springer-Verlag, 1994. p. 1-25.

USCHOLD, M.; KING, M. *Towards a methodology for building ontologies*. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 1995, Canada. Disponível em: < citeseer.ist.psu.edu/uschold95toward.html >. Acesso em: 10 set. 2004.

VON BREVERN, H. Cognitive and Logical Rationales for e-Learning Objects. *Educational Technology & Society*, v. 7, n. 4, p. 2-25, 2004.

GLOSSÁRIO

As definições a seguir relacionadas foram extraídas das referências, especialmente de Larman (2007), Russel e Novig (2004), Cimiano (2006) e OMG (2007).

Abstração – ato de se concentrar nas qualidades essenciais ou gerais de coisas semelhantes.

Agregação – uma propriedade de uma associação que representa um relacionamento todo-parte.

Aprendizagem de máquinas - tem como objetivo aprender regras de extração a partir de um conjunto (*corpus*) de treinamento; é equivalente à construção de regras baseadas em observações de estados e transições ambientais. O comportamento das regras de extração geradas depende da seleção manual dos exemplos de treinamento. Um algoritmo de aprendizagem utiliza o conjunto de dados e a informação que o acompanha como entrada e retorna uma declaração - ou seja, um conceito representando os resultados da aprendizagem como saída. Assim examina os exemplos anteriores e suas saídas, aprende a reproduzi-los e faz generalizações sobre novos casos. Pode ser Aprendizado Supervisionado ou Não-supervisionado.

Aprendizagem de máquinas supervisionado - aquele no qual o algoritmo de aprendizado (indutor) recebe um conjunto de exemplos de treinamento para os quais os rótulos da classe associada são conhecidos.

Aprendizagem de máquinas não-supervisionado - aquele no qual o indutor analisa os exemplos fornecidos e tenta determinar se alguns deles podem ser agrupados de alguma maneira, formando agrupamentos ou clusters.

Anotação - uma nota explicativa ou crítica anexada ao recurso.

Associação – uma descrição de um conjunto relacionado de ligações entre objetos de duas classes.

Associação recursiva – uma associação na qual a origem e o destino são a mesma classe de objetos.

Axiomas - são sentenças lógicas sempre verdadeiras. Axiomas lógicos (proposições) representam o conhecimento e regras de dedução (inferência) de novos conhecimentos.

Cardinalidade - representa o número exato de valores que devem estar associados à um atributo para uma classe.

Ciclo de vida da ontologia – todos os aspectos ligados a cada processo relacionado a representação de ontologias, como criação, manutenção, avaliação e uso.

Classes disjuntas - o indivíduo que é membro de uma classe não pode ser membro de outra classe simultaneamente.

Classificador abstrato – em UML, é um elemento de modelo que descreve características comportamentais e estruturais.

Cláusulas de Horn - é uma cláusula (disjunção de literais) com no máximo um literal *positivo*. Representação de proposições na forma de uma sentença “se p então q ”, ou seja, $q \leftarrow p$ ou $p \rightarrow q$.

Clusterização - consiste em organizar conceitos automaticamente em grupos de conceitos similares, gerando assim facilidades para sua visualização, manipulação e análise.

Complemento - a operação de complemento sobre conjuntos é definida em relação a um conjunto fixo U denominado *conjunto universo*, ou seja, $A' = \{ x \mid x \in U \text{ e } x \notin A \}$.

Construção acelerada de ontologias - construção de ontologias a partir de técnicas de aprendizagem de ontologias tendo como resultado um tempo menor na elaboração da representação das ontologias.

Descrição Lógica – é um formalismo resultante de um subconjunto da Lógica de Primeira Ordem utilizado para representar um domínio de forma estruturada e com semântica bem definida.

Disjunção - sejam p e q duas proposições quaisquer; a partir delas pode-se construir uma nova proposição, designada por $p \vee q$ (leia-se “ p ou q ”), chamada disjunção de p com q . Duas ou mais categorias são disjuntas se elas não têm elementos em comum.

Domínio ontológico - um campo particular do conhecimento representado numa ontologia.

Domínio - um campo particular do conhecimento ou um limite formal que define um assunto ou área de interesse em particular.

Engenharia progressiva - é o processo tradicional de produzir software a partir de abstrações de alto nível, através de projetos independentes de implementação, gerando a versão física do sistema.

Engenharia de Ontologia – reúne as linguagens e ferramentas para construção de ontologias e adiciona um conjunto de atividades importantes para o processo de desenvolvimento de ontologias e para o ciclo de vida da ontologia (PINTO; TEMPICH; SURE, 2004).

Engenharia reversa - é uma coleção de teorias, metodologias e técnicas capazes de apoiar a extração e abstração de informações de um software existente, produzindo documentos consistentes, quer seja a partir do código fonte, ou através da adição de conhecimento e experiência que não podem ser, automaticamente, reconstruídos a partir do código.

Framework - pode ser definido como um conjunto de artefatos de software selecionados com o objetivo de apresentar uma solução num determinado domínio de problema. Estes artefatos podem ser usados, estendidos ou personalizados por desenvolvedores para obtenção de soluções específicas. Os *frameworks* constituem um avanço em termos de reuso de software, uma vez que esses ocorrem não apenas a partir de pequenos blocos de programas, mas sim de todo ou parte de um sistema.

Ferramenta de Autoria – é um software aplicativo que permite aos usuários criarem conteúdo educacional em formato eletrônico – objetos de aprendizagem, projeto de aprendizagem etc. – como os citados no Quadro D1.

Ferramenta de Aprendizagem – é definida como qualquer objeto, exercício ou experiência utilizada pelo estudante na consecução de objetivos educacionais como o uso do computador com fóruns de discussões e plataformas de EAD.

Generalização – a atividade de identificar características comuns entre conceitos e definir relacionamentos de uma superclasse – conceito geral – e de uma subclasse – um conceito especializado; torna o modelo de alguma experiência específica, em uma experiência geral; é uma maneira de construir classificações taxonômicas entre conceitos, que são então ilustrados em hierarquias de classes.

Grau de incerteza – as incertezas têm sido tratadas nas teorias da probabilidade, da possibilidade, da evidência, dos subconjuntos *fuzzy*, e na teoria clássica dos conjuntos; está relacionada a inabilidade para discriminar qual, entre várias alternativas, é a real, ou verdadeira, em uma dada situação pela existência de algum tipo de discrepância ou inconsistência na informação ou evidência. A incerteza pode então, ser caracterizada pelos vetores de probabilidade que são produzidos como um subproduto da maioria dos algoritmos de classificação. Estas probabilidades fornecem uma informação útil sobre a qualidade da classificação resultante em termos das incertezas envolvidas.

Grau de participação – pode-se classificar os elementos considerados, não mais como pertencentes ou não a um conjunto, mas sim de acordo com seu respectivo

grau de pertinência ao conjunto estudado. Este grau assume valores no intervalo [0 a 1], cujos extremos representam, respectivamente, o conceito da não pertinência e da pertinência plena. Estes conjuntos permitem que se definam operações binárias internas, tais como união, interseção e complemento.

Indivíduos - representam objetos no domínio de interesse; são também conhecidos como instâncias e são referenciados como 'instâncias de classes'.

Herança – uma característica pela qual as classes podem ser especializadas a partir de superclasses mais gerais. Os atributos e definições de método das superclasses são adquiridos automaticamente pela subclasse.

Hierarquia de classificação - pode ser vista através de níveis de classificação, iniciando com um objeto elementar (folha) até o objeto máximo da hierarquia de classificação (raiz).

Learning Object (LO) – ver Objeto de aprendizagem.

Máquina de inferência Bossam - sistema que implementa um mecanismo de inferência com um motor de regras baseado no algoritmo RETE (armazena a informação sobre as regras, na rede e ao invés de buscar as semelhanças dos fatos em todas as regras em todos os ciclos de reconhecimento das ações, somente procura por trocas semelhantes em todos os ciclos). Suporta interação com as linguagens OWL, SWRL e RuleML.

Meta-metamodelo – um metamodelo que define outros metamodelos.

Metamodelo – um modelo que define e descreve toda a estrutura conceitual sintática e semântica dos diferentes modelos que o compõem e permite criar mecanismos para verificar a consistência dos/entre seus diferentes modelos.

Modelo – uma descrição das características estáticas e/ou dinâmicas de uma área de interesse, retratada por meio de diversas visões.

Modelo de Estudante – contem informações sobre comportamento, desempenho e aprendizado do estudante durante sua interação com o ambiente educacional.

Objeto de Aprendizagem – Para o *Learning Technology Standards Committee* (LTSC) do IEEE, um objeto de aprendizagem é: "qualquer entidade, digital ou não digital, que possa ser usada para aprendizado, educação e treinamento."

Padrão – uma descrição nomeada de um problema, uma solução, de quando aplicar a solução e como aplicá-la em novos contextos.

Pattern matching – é o ato de verificação da presença de um padrão em um conjunto de dados.

População da ontologia – inserir/obter um conjunto de instâncias na ontologia definida.

Projeto de Aprendizagem (PA) – é uma aplicação concreta de um modelo pedagógico para um grupo específico, com objetivos específicos e num domínio específico. O PA especifica o fluxo de trabalho e conteúdo necessário para o processo de ensino e aprendizagem: quais papéis realizam quais atividades, usam-se quais recursos e serviços, em qual ordem eles são aplicados para atingir os objetivos de aprendizagem, considerando a melhor maneira e as diferenças individuais.

Propriedade de objeto – em UML, é um valor nomeado que denota uma característica de um elemento. Uma propriedade tem impacto semântico.

Propriedade equivalente – uma propriedade é equivalente se possuir as propriedades reflexiva, transitiva e simétrica.

Propriedade funcional – uma propriedade funcional R é aquela na qual se tem um único valor de y para cada instância em x , ou, para cada x e W , existe um único $y \in W$, tal que $R(x,y)$.

Propriedade inversa - as propriedades têm um sentido; no caso da inversa, este sentido pode ser dado de ambos os lados; assim, para cada par $P1(x,y)$, existe um par inverso $P2(y,x)$.

Propriedade reflexiva - uma propriedade R é reflexiva quando todo elemento de E se relaciona com ele mesmo, ou, para cada $x \in E$, existe $R(x,x)$ em E . Todo conjunto é subconjunto de si próprio.

Propriedade simétrica - uma propriedade R é simétrica quando, se x se relaciona com y , y se relaciona com x , ou, para cada $x, y \in E$; $R(x,y)$ implica $R(y,x)$.

Propriedade transitiva - uma propriedade R é transitiva quando, se x está relacionado com y e y está relacionado com z , então x se relaciona com z , ou, para cada $x, y, z \in E$, existe $R(x,y)$ e $R(y,z)$ implica $R(x,z)$. Se um conjunto é subconjunto de um outro e este é subconjunto de um terceiro, então o primeiro é subconjunto do terceiro.

Prova de conceito - verificar a existência de provas de conceito da tecnologia, ou seja, analisar os resultados obtidos em um ambiente de laboratório projetado para simular o ambiente real.

Recurso - na Web, o conceito recurso significa qualquer objeto que pode ser atingido através de uma URI, ou seja, qualquer recurso acessado via seu endereço eletrônico.

Relacionamentos hierárquicos - possuem três subtipos possíveis - genérico – que representa relacionamentos tipo classe/subclasse, instância – que representa relacionamentos do tipo classe/instância e todo/parte – que representa relacionamentos nos quais um conceito é parte de outro.

Relação *is-a* - relação hierárquica entre conceitos onde o predicado é “é-um” – expressa hierarquia.

Relação *part-of* - relação hierárquica entre conceitos de classificação parte-de (classe e subclasse); alguma coisa faz parte de outra.

Relação ação - relação entre conceitos onde o predicado expressa uma ação.

Relação mereológica – idem relação *part-of*.

Relação propósito - relação não-taxonômica entre conceitos onde o predicado expressa um propósito.

Relação de causa - relação entre conceitos onde o predicado expressa uma consequência.

Relação de atributo - relação entre conceitos onde o primeiro conceito é um atributo do segundo conceito.

Relação não-taxonômica – é uma relação entre conceitos que denota uma ação.

Relação taxonômica – relação entre conceitos com objetivo de classificar informações – seus assuntos, naturezas, tipos etc., expressos numa hierarquia.

Restrições - referem-se aos valores possíveis dos atributos dos conceitos numa ontologia. O uso de axiomas restringe ao máximo a interpretação sob um determinado atributo. Correspondem - as assinaturas de classes no modelo OO, aos axiomas universalmente quantificados em modelos lógicos, as restrições de integridades nos esquemas de banco de dados.

Schemata - conjunto de conhecimento representado com sintaxe e semântica formais, explícitas e estruturadas de acordo com a técnica que implementam.

Técnicas de aprendizagem de ontologias - qualquer técnica expressa em algoritmos ou regras, empregada para obtenção de ontologias a partir de recursos existentes.

Transformação de modelos - é a geração automática de um modelo destino a partir de um modelo origem, definida por um conjunto de regras que juntas descrevem como um modelo na linguagem origem pode ser transformado em um ou mais modelos na linguagem destino, pode ocorrer entre mesmos níveis e entre diferentes níveis de abstração.

XMI - uma linguagem da OMG que define um conjunto de regras para mapear meta-metamodelo, metamodelos e modelos para documentos XML. Os documentos XML têm o objetivo de prover, de maneira simples e independente de plataforma, a interoperabilidade através do uso de cadeias de textos dotadas de descrições. XMI utiliza a *Document Type Definition* (DTD) para validar os documentos XML de acordo com os seus respectivos metamodelos. A DTD define a estrutura dos elementos que podem ser descritos nos documentos XML

UML profiles - é um sub-conjunto especializado da especificação da UML que define como elementos do modelo UML são personalizados e estendidos, adicionando nova semântica, através do uso de quatro mecanismos: estereótipos, definição de marcação (tag), valores atribuídos e restrições.

ANEXO A – Ferramentas

Este anexo apresenta quadros com as principais ferramentas vistas durante o desenvolvimento do trabalho.

Ferramenta	Referência
Amaya	http://www.w3.org/Amaya/
Mangrove	http://www.cs.washington.edu/research/semweb/
Vannotea	http://www.itee.uq.edu.au/~ereasearch/projects/vannotea/index.html
OntoMat	http://annotation.semanticweb.org/ontomat/index.html
OntoAnnotate ou M-OntoMat-Annotizer	http://www.acemedia.org/aceMedia/results/software/m-ontomat-annotizer.html
SHOE (<i>Simple HTML Ontology Extension</i>)	http://www.cs.umd.edu/projects/plus/SHOE/KnowledgeAnnotator.html
SMORE (<i>Semantic Markup, Ontology, and RDF Editor</i>)	http://www.mindswap.org/2005/SMORE/
OOF (<i>Open Ontology Forge</i>)	http://research.nii.ac.jp/~collier/resources/OOF/index.html
COHSE (<i>Conceptual Open Hypermedia Services Environment</i>)	http://cohse.cs.manchester.ac.uk/

Quadro A1 - Ferramentas para Anotação Manual

Ferramenta	Referência
Lixto	http://www.dbai.tuwien.ac.at/proj/lixta/
MnM	http://kmi.open.ac.uk/projects/akt/MnM/
Melita	http://nlp.shef.ac.uk/melita/
Parmenides	http://www.ifi.uzh.ch/cl/Parmenides/
Armadillo	http://www.x-media-project.org/X-media%20Project/armadillo.ht
KnowItAll	http://www.cs.washington.edu/research/knowitall/
SmartWeb	http://smartweb.dfki.de/
PANKOW (<i>Pattern-based ANnotation Knowledge through On the Web</i>)	http://km.aifb.uni-karlsruhe.de/pankow/
AeroSWARM (<i>AeroText Semantic Web Automated Relation Markup</i>)	http://projects.semwebcentral.org/projects/aeroswarm/
SemTag	http://www2003.org/cdrom/papers/refereed/p831/p831-dill.html
KIM (<i>Knowledge and Information Management</i>)	http://www.ontotext.com/kim/introduction.html
RAINBOW (<i>Reusable Architecture for Intelligent Brokering Of Web</i>)	http://rainbow.vse.cz/
h-TechSight	http://prise-serv.cpe.surrey.ac.uk/techsight/
S-CREAM (<i>Semi-automatic CREAtion of Metadata</i>)	http://de.scientificcommons.org/8082936

Quadro 2 - Ferramentas para Anotação Automática

Ferramenta	Referência
WickOffice (<i>Writing in the Context of Knowledge Office</i>)	(CARR et al.; 2004)
AktivDoc	(LANFRANCHI; CIRAVEGNA; PETRELLI; 2005)
SemanticWord	(TALLIS; 2003)
Magpie	(DZBOR; MOTTA; DOMINGUE; 2004)
Thresher	(HOGUE; KARGER; 2005)

Quadro A3 - Ferramentas para Anotação Integrada

Linguagem / Descrição	Referência
CycL	http://www.cyc.com/cycdoc/ref/cycl-syntax.html
Ontolingua	http://ontolingua.stanford.edu/
F-Logic <i>Frame Logic</i>	http://www.ontoprise.de/documents/tutorial_flogic.pdf
CML <i>Conceptual Modelling Language</i>	http://hcs.science.uva.nl/projects/void/cml.html
OIL <i>Ontology Inference Layer</i>	http://www.ontoknowledge.org/oil/
DAML+OIL <i>DARPA Agent Markup Language</i>	http://www.daml.org
RDF e RDFS	http://www.w3.org/RDF/
XOL <i>XML-Based Ontology Exchange Language</i>	http://www.ai.sri.com/pkarp/xol/
SHOE	http://www.cs.umd.edu/projects/plus/SHOE/
XTM <i>XML Topic Maps</i>	http://www.topicmaps.org/xtm/1.0
OWL	http://www.w3.org/2004/OWL
PowerLoom	http://www.isi.edu/isd/LOOM/PowerLoom/index.html
FOML <i>Formal Ontology Markup Language</i>	http://citeseer.ist.psu.edu/544205.html
NKRL <i>Narrative Knowledge Representation Language</i>	http://citeseer.ist.psu.edu/694933.html
OML <i>Ontology Markup Language</i>	http://xml.coverpages.org/oml.html
OCML <i>Operational Conceptual Modeling Language</i>	http://kmi.open.ac.uk/projects/ocml/

Quadro A4 - Linguagens para representação de ontologias

Linguagem / Descrição	Referência
RDQL <i>RDF Data Query Language</i>	http://www.daml.org/tools/
SWRL	http:// www.w3.org/Submission/SWRL/
nRQL <i>new RACER Query Language</i>	http://www.racer-systems.com/products/racerpro/features.phtml
OWL-QL <i>OWL Query Language</i>	http://ksl.stanford.edu/projects/owl-ql/
<i>Vampire</i>	http://frdcsa.onshore.net/frdcsa/binary/vampire-7.0/
SPARQL <i>Simple Protocol and RDF Query Language</i>	http://www.w3.org/TR/2006/WD-rdf-sparql-query-20061004/

Quadro A5 - Linguagens para consultas de ontologias

ANEXO B - *Ontology Definition Metamodel*

Os elementos de ontologia – conceitos, indivíduos, relações, funções e axiomas – encontram representação em vários modelos de linguagens como RDFS, OWL, SHOE, CycL, SPARQL entre outras. Assim, o ambiente de ontologias tornou-se heterogêneo e padrões foram definidos para fundamentar a representação, o gerenciamento e a interoperabilidade das aplicações com ontologias. W3C padronizou a representação com as linguagens RDFS e OWL e a consulta com SWRL. OMG padronizou o metamodelo dos modelos de linguagem com o *Ontology Definition Metamodel* (ODM) para garantir o gerenciamento e interoperabilidade. Este metamodelo surgiu em 2003, quando as empresas IBM e Sandpiper submeteram-no à aprovação do OMG e encontra-se na sexta revisão (OMG, 2007a). Os principais objetivos do ODM são:

- fundamentar conceitos para utilização de técnicas familiares, como UML, na representação formal de ontologias com lógica descritiva, como OWL/DL, e lógica de primeira ordem, como OWL *Full*. Para isso é necessário aumentar o nível de expressividade e complexidade das técnicas existentes;
- garantir consistência, validação e conferência dos mecanismos de raciocínio baseados numa lógica formal padronizada;
- apoiar as atividades do ciclo de vida da ontologia como análise conceitual, captura, representação e manutenção do conhecimento;
- permitir o desenvolvimento de mecanismos para avaliar a qualidade da ontologia.

Ontologias representadas com linguagens suportadas por este metamodelo trocam representação do conhecimento entre sistemas heterogêneos e especificação de expressões que ora são a entrada para, ou a saída de, mecanismos de inferência.

A UML participa do ODM sob dois aspectos: um para representá-lo e outro para servir como notação na modelagem de ontologias. Neste segundo aspecto, logo surgem elementos definidos numa representação e não presentes noutra, ou vice-versa. Para resolver isto, existem os *Profile* UML - um conceito usado para adaptação dos elementos básicos da UML para um objetivo específico. Significa introduzir novos elementos de modelagem e adicioná-los no repertório de ferramentas. Os elementos da UML podem ser personalizados e estendidos pelo

uso de quatro mecanismos definidos na especificação da UML: estereótipos, definição de marcação (tag), valores atribuídos e restrições.

Estereótipos garantem uma definição virtual de subclasses das metaclasses UML através do assinalamento de uma semântica adicional. Por exemplo, define-se o estereótipo <<OntClass>> para simbolizar o elemento de modelagem usado para representar as classes da ontologia.

Definição de marcação introduz novas propriedades para os elementos do modelo e são análogas às definições de meta-atributos. Cada definição de marcação especifica o valor atual das propriedades dos elementos de um modelo individual. Por exemplo, o estereótipo <<OntClass>> pode ter valores atribuídos para seus meta-atributos como complemento, enumeração, intersecção, união.

Restrições garantem o refinamento semântico dos elementos de modelagem que foram incluídos. Podem ser atribuídas ao estereótipo usando OCL (*Object Constraint Language*). Por exemplo, para os meta-atributos do estereótipo <<OntClass>>, somente um valor poderá ser atribuído para um meta-atributo, ou para intersecção, ou para complemento, ou para enumeração, ou para intersecção.

Assim, para ODM atingir seus objetivos, primeiro apresentam-se os metamodelos dos modelos das linguagens consideradas, depois os *profiles* UML necessários para representar estes modelos, seguidos pelos mapeamentos entre os metamodelos.

Os metamodelos das representações considerados foram:

- dois metamodelos para representarem as linguagens da lógica formal: DL (*Description Logics*) e CL (*Common Logic*);
- três metamodelos para representarem as linguagens RDFS, OWL e *Topic Maps* - menos expressivas do que DL e CL;
- dois metamodelos para representarem a UML versão 2 (UML2) e E/R.

Os três *Profiles* UML identificados como necessários para garantir o uso da notação UML na modelagem de ontologias e para facilitar a geração da sua representação em RDFS, OWL e *Topic Maps*, são: ODM para RDF, ODM para OWL e ODM para *Topic Maps*.

Os mapeamentos no ODM são expressos na linguagem de relações denominada MOF QVT (*Query View Transformations*). ODM inclui definições para o mapeamento direto para e de OWL, para UML e *Topic Maps*.

Para este trabalho destacam-se a representação do metamodelo OWL, o *profile* UML para OWL e o mapeamento UML para OWL.

O metamodelo OWL tem vários diagramas para representar a totalidade do modelo teórico semântico da OWL em conformidade com o metamodelo MOF.

A Figura A1 mostra um fragmento do metamodelo com o diagrama de classes UML representando o elemento *OWLClass* da OWL como uma classe. Todas as funções de complemento, intersecção e união da OWL surgem como generalizações da classe *OWLClass*, herdando suas propriedades, e sendo denominadas *ComplementClass*, *IntersectionClass* e *UnionClass*, respectivamente. Já as instâncias da OWL ficam representadas na classe de enumeração, denominada *EnumeratedClass*, que também herda as características da classe *OWLClass*. Já as funções de equivalência e disjunção surgem como associações da *OWLClass* com ela mesma denominadas *OWLequivalentClass* e *OWLdisjointClass*.

Figura B1 – Diagrama da Descrição da Classe OWL (OMG, 2007a)

Como as restrições representadas na OWL para suas classes são muitas, de acordo com as funções consideradas para elas, a classe *OWLRestriction*, presente na Figura B1, foi detalhada no diagrama mostrado na Figura B2 também como generalização – *HasValueRestriction*, *CardinalityRestriction*, *AllValueFromRestriction*, *SomeValueRestriction*, *MaxCardinalityRestriction* e *MinCardinalityRestriction*.

Figura B2 – Diagrama das Restrições de Classes OWL (OMG, 2007a)

Como as restrições em OWL servem também para as propriedades, a seta de navegabilidade apontando da classe fonte *OWLRestriction* para a classe alvo *RDFProperty* indica isto - uma classe *OWLRestriction* tem um atributo de uma *RDFProperty*. A visibilidade do atributo é dada por *OWLonProperty* e a multiplicidade pelo número 1. Esta interpretação é válida para todas as setas de navegabilidade. E assim um conjunto de diagramas UML representa o metamodelo da OWL com seus elementos.

Agora para a UML ser utilizada como notação para modelar ontologias é necessário definir um *Profile* UML com as características UML sem equivalência na OWL e vice-versa.

As características UML sem equivalência na OWL são as comportamentais -como para calcular valores das propriedades, operações para descrever os parâmetros dos métodos, operações estáticas, classes de interface e de controle - e as relativas ao controle de acesso às propriedades - classes tem elementos públicos ou privados, subclasses de metaclasses, entre outras.

As características OWL sem equivalência na UML são propriedades globais, classes como instâncias, restrições só para uma dada classe (*allValuesFrom*, *someValuesFrom*), propriedades simétricas e transitivas (*SymmetricProperty*, *TransitiveProperty*), classes disjuntas (*disjointWith*), operação de complemento (*complementOf*), entre outras.

Uma vez representado o metamodelo OWL e, tendo presente os elementos do metamodelo UML, foi definido o *profile* correspondente, ou seja, realizado a

correspondência entre os elementos comuns, os inseridos e os elementos sem equivalência.

A Figura B3 mostra um exemplo de como ficou a representação da propriedade *disjointWith* no diagrama UML: as associações de generalização aliada a linha tracejada significam *disjointWith*.

Figura B3 – Exemplo Estereótipo

Alguns elementos não encontram correspondência ou não podem ser expressos em estereótipos. Por exemplo, OWL permite uma subclasse ser declarada usando *subClassOf* ou ser inferida a partir da definição de uma classe relacionada com outras classes. Isto permite a definição de uma classe a partir de um conjunto de indivíduos que satisfaçam uma restrição. Esta expressão pode ser uma combinação booleana de outras classes (*intersectionOf*, *unionOf*, *complementOf*), ou uma restrição no valor de uma propriedade (*hasValue*). A OCL ou CL são utilizadas para representar estas e outras situações.

Agora com os elementos identificados nos dois metamodelos é necessário definir as regras para mapeamento entre eles. Na Figura B4 observa-se um exemplo simples de mapeamento das propriedades de duas classes *Course* e *Student*.

Figura B4 – Exemplo de Mapeamento UML para OWL

Assim o modelo UML construído para representar uma ontologia em alguma ferramenta de projeto, como para modelagem conceitual, pode ser exportado para outra ferramenta de projeto, como para edição, via um documento XMI – arquivo serializado do metamodelo. Antes da importação na ferramenta destino, o documento XMI pode passar por ferramentas de transformação ou validação. A Figura B5 apresenta esta situação.

Figura B5 – Esquema Genérico de Transporte de Modelos

As ferramentas de transformação consideram o elemento do modelo da linguagem de origem, o metamodelo da linguagem de origem, o elemento do metamodelo de destino e a regra aplicada no mapeamento. O Quadro B1 exemplifica estas considerações, respectivamente (DOA N et al., 2002).

OWL	Metamodelo OWL	Metamodelo UML	Regra
OneOf	owl:OneOf	EnumeratedClass	Disjunção
maxcardinality	owl:maxcardinality	Restriction MaxCardinality	Multiplicidade da cardinalidade

Quadro B1 – Exemplo de Transformação entre metamodelos

ODM está dentro do contexto MDA e insere os elementos da WS no contexto da Engenharia de Software.

ANEXO C – Métodos e Técnicas em AO

Os métodos e técnicas construídas e aplicadas no AO a partir de textos envolvem as áreas de Processamento de Linguagem Natural (PLN), Análise Formal de Conceitos e Aprendizagem de máquinas - todas extremamente complexas e interdisciplinares (NAVIGLI; VELARDI, 2004; BIEMANN, 2005).

PLN é uma subárea da Inteligência Artificial e da Lingüística para estudo dos problemas da geração e compreensão automática de um idioma. Os sistemas construídos convertem informação representada em alguma linguagem computacional (HTML, SQL etc.) em linguagem normalmente compreensível ao ser humano, ou convertem ocorrências de linguagem humana em representações mais formais, mais facilmente manipuláveis por programas de computador. Estes sistemas precisam reconhecer o contexto, fazer análise morfológica, sintática, semântica e léxica, extrair informação, interpretar os sentidos e até aprender conceitos com os textos processados.

A análise morfológica identifica palavras ou expressões isoladas em uma sentença (*stemming*), sendo este processo auxiliado por delimitadores como pontuação e espaços em branco. As palavras identificadas são classificadas de acordo com seu tipo de uso ou, em linguagem natural, categoria gramatical.

Neste contexto, uma instância de uma palavra em uma sentença gramaticalmente válida pode ser substituída por outra do mesmo tipo, configurando uma sentença ainda válida com substantivos, pronomes, verbos, entre outros. Dentro de um mesmo tipo de palavra, existem grupos de regras que caracterizam o comportamento de um subconjunto de vocábulos da linguagem como formação do plural de substantivos (como os terminados em "ão"), flexões dos verbos regulares (como os terminados em "ar"), entre outros. Assim, a morfologia trata as palavras quanto à sua estrutura, forma, flexão e classificação, no que se refere a cada um dos tipos de palavras.

Através da gramática da linguagem a ser analisada e das informações da análise morfológica, a análise sintática procura construir árvores de derivação para cada sentença, mostrando como as palavras se relacionam.

Durante a construção da árvore de derivação, é verificada a adequação das seqüências de palavras às regras de construção impostas pela linguagem, na

composição de frases, períodos ou orações. Dentre estas regras, pode-se citar a concordância e a regência nominal e/ou verbal, bem como o posicionamento de termos na frase.

Nos sistemas de PLN, o maior problema é a transformação de uma frase potencialmente ambígua em uma não ambígua, a qual será utilizada pelo sistema. Esta transformação é conhecida como *parsing*.

Pode-se dizer que a semântica se desdobra em semântica léxica e em semântica gramatical. A semântica léxica busca uma representação conceitual para descrever o sentido, sendo que, para construir esta representação, pode ser feita a decomposição semântica das unidades léxicas, ou podem ser utilizadas redes semânticas.

À medida que se avança, é necessário fazer uma interpretação do todo e não mais analisar o significado de suas partes, do ponto de vista léxico e gramatical. As ligações anafóricas como as pronominais - ele, este etc. - e de outros fenômenos de referência merecem resolução.

Algumas técnicas utilizadas no contexto PLN são: tokenização, normalização, fronteiras de frases, etiquetagem, padrões gramaticais, reconhecimento de entidade, análise dos constituintes, co-referência e detecção automática de sinônimos⁸⁰. Além dessas possibilidades, as relações semânticas entre os termos lingüísticos podem ser estabelecidas por métodos de agrupamento e classificação.

Alguns modelos definidos aliam caráter estatístico e probabilístico cuja essência do processo é a identificação automática de palavras-chave no texto pela frequência com que aparecem. Destacando a indexação estatística de termos por frequência, a TFIDF (*Term frequency, inverse document frequency*), o método *N-grams*, e os *Stemmers* (extraem a raiz das palavras).

Análise Formal de Conceito, ou *Formal Concept Analysis*⁸¹, é a teoria de análise de dados estruturados, como conceitos formais, para concepção e análise de hierarquias de conceitos. As técnicas utilizadas tentam descobrir relacionamento entre dados. Descoberto um relacionamento, este pode ser expresso através de regras. Os reticulados conceituais permitem a extração de regras determinísticas e probabilísticas. O conjunto de regras determinísticas é formado por regras de implicações e dependências funcionais. As regras probabilísticas dividem-se em

⁸⁰ http://www.maxwell.lambda.ele.puc-rio.br/cgi-bin/PRG_0599.EXE/10081_5.PDF?NrOcoSis=32308&CdLinPrg=pt

⁸¹ <http://www.upriss.org.uk/fca/fca.html>

regras de associação e de classificação. Esta última indica atributos que caracterizam uma classe.

Aprendizagem de Máquinas tem como objetivo aprender regras de extração a partir de um conjunto (*corpus*) de treinamento. O comportamento das regras de extração geradas depende da seleção manual dos exemplos de treinamento. Pode ser Aprendizado Supervisionado ou Não-supervisionado.

O Aprendizado Supervisionado é aquele no qual o algoritmo de aprendizado (indutor) recebe um conjunto de exemplos de treinamento para os quais os rótulos da classe associada são conhecidos. Cada exemplo - instância ou padrão - é descrito por um vetor de valores (atributos) e pelo rótulo da classe associada. O objetivo do indutor é construir um classificador que possa determinar corretamente a classe de novos exemplos ainda não rotulados. Para rótulos de classe discretos, esse problema é chamado de classificação e para valores contínuos como regressão.

O Aprendizado Não-Supervisionado é aquele no qual o indutor analisa os exemplos fornecidos e tenta determinar se alguns deles podem ser agrupados de alguma maneira, formando agrupamentos ou clusters. Após a determinação dos agrupamentos, em geral, é necessário uma análise para determinar o que cada agrupamento significa no contexto do problema que está sendo analisado.

Algumas técnicas utilizadas no aprendizado de máquinas são *k*-NN, *Naive Bayesian Learning*, árvores de decisão, Redes Neurais, *Support Vector Machines*, *Ensembles*, Regras de Associação, *k-means* e Métodos de agrupamento hierárquico.

Os métodos e aplicações para AO utilizam estas abordagens (PLN, Análise Formal de Conceitos e Aprendizado de Máquinas) com algoritmos específicos para cada intenção de recuperação (termo, sinônimo, conceito, relação, axioma).

Cimiano (2006) realizou um amplo estudo sobre os métodos e as técnicas aplicadas na AO a partir de textos, e também relacionou os principais projetos e estudos existentes. Tudo de acordo com sua camada de elementos da ontologia. O Quadro C1 sintetiza o resultado deste estudo.

Elemento	Técnica	Comentário
Termos	Indexação de termos (PLN) + processamento estatístico	Etiquetagem (<i>part-of-speech</i>) (categorias léxico sintáticas, conjunto completo de rótulos,...)
	Lingüísticos	Análise Linguística – ☐ <i>Part-of-Speech Tagging</i> , Análise morfológica, Reconhecimento frases Extração de padrões ☐ Adj-Nome, Nome-Nome, Adj-Nome-Nome, Adj*Nome*
	Estatísticos	Análise de co-ocorrência para extração de termos com corpus Comparação de frequências entre domínio e corpora
	Híbridos	Regras lingüísticas Filtros estatísticos (pré- ou post-)
Sinônimos	Clusterização (similaridade semântica)	LSI (<i>Latent Semantic Indexing</i>) PLSI (<i>Probabilistics LSI</i>)
		Busca por termos compartilham mesma semântica Busca por termos com mesmo significado Tradução entre linguagens ☐ Dicionários ☐ Corpora comparável (bilingual)
Conceitos	Redução da dimensão Ponto de vista extensional Intencional	LSI Know-It-All OntoLearn

Quadro C1 – Resumo Elemento versus Técnica Utilizada no AO

Elemento	Técnica	Comentário
Conceitos	Termo => conceito	Intenção – definição formal ou informal do conjunto de objetos que o conceito descreve
	Reconhecimento entidades nomeadas	Extensão – conjunto de objetos (instâncias) que a definição deste conceito descreve; extração das instâncias do texto; relacionar com marcações (metadados);
	Extração de informação	Realização léxica – os sinônimos
	Classificação	Heurística, Wordnet, padrões
Hierarquia de conceitos	Padrões léxico-sintáticos Hipótese distribucional de Harris Análise de co-ocorrência de termos	Criação de algoritmos para clusterização – grupos de palavras similares
Relações	Co-ocorrência de termos Dependência sintática	Algoritmos com regras de associação Adquirir restrições para verbos
Taxonomia	Padrões léxico-sintáticos	<i>Hearst Patterns</i>
	Clusterização de hierarquia de conceitos e distribuição por similaridade	Baseado na similaridade (Ex.: KMean) Probabilidade do termo ocorrer junto com outros no conjunto <i>Soft clustering</i>
	Abordagens lingüísticas	
	Document-subsumption	
	Extensão / refinamento	LSI, Assinaturas, Tree-ascending + kNN, árvores de decisão
Axiomas	Esquema axiomas incoerentes (disjoint) Expressões condicionais quantificadas Programação Lógica Indutiva	Se termos aparecem coordenados numa expressão como “homem e mulher” então eles são incoerentes Para derivar cláusulas de Horn

Quadro C1 – Resumo Elemento versus Técnica Utilizada no AO (continuação)

ANEXO D – Padrões de Projeto de Aprendizagem

Koper (2004) definiu projeto de aprendizagem como uma aplicação de um modelo pedagógico para um objetivo de aprendizagem específico, um grupo alvo, e um contexto específico ou domínio de conhecimento. Uma parte importante desta definição é a abstração dos conceitos da pedagogia do contexto e conteúdo; assim, modelos pedagógicos podem ser compartilhados e reutilizados entre contextos instrucionais e matérias do domínio.

O IMS LD é um modelo para especificação de atividades de aprendizagem com origem no EML (*Educational Modeling Language*) da OUNL.

Os estudos do EML iniciaram-se em 1998 com o objetivo de criar um modelo conceitual para representar uma unidade de aprendizagem em sua totalidade, englobando não só o conteúdo como também as diversas atividades envolvidas (KOPER et al., 2002). Assim, o modelo atende a descrição de uma ampla variedade de processos de aprendizagem - cursos virtuais e presenciais, programas de estudo, apostilas, livros etc. - representando, não apenas seu conteúdo - textos, tarefas, provas etc. -, mas também as regras, relações, interações e atividades dos estudantes, professores e monitores. Este modelo foi representado com os diagramas da UML, um padrão reconhecido para representação de artefatos dos sistemas. A codificação do EML é em XML, uma meta-linguagem internacionalmente aceita para a estruturação e descrição de documentos e dados, sendo o Edubox seu primeiro resultado prático.

As principais especificações EML são as seguintes (AMORIM; LAMA; VILA, 2006):

- A-CDF⁸² (*ARIADNE Course Description Format*) é uma especificação do consórcio ARIADNE para a descrição de cursos. Um curso em A-CDF é uma unidade de estudo e consiste de documentos XML com um gerador de curso LMS. Ele coloca especial ênfase no conteúdo e sua agregação. O material didático gerenciado por ele é restrito ao formato texto e ele usa a combinação de ferramentas desenvolvidas pelo consórcio (editores de currículos, LMS, KPS (*Knowledge Pool System*)).

⁸² <http://www.ariadne-eu.org>

- LMML⁸³ (*Learning Material Mark-up Language*) descreve o material de aprendizagem em XML e abrange vários módulos com material de aprendizagem, cada qual contendo outros sub-módulos.
- PALO⁸⁴ é uma especificação desenvolvida pela Universidade Nacional de Ensino a Distância da Espanha. Descreve cursos organizados em módulos contendo atividades de aprendizagem, conteúdo e um plano de ensino associado.
- Targeteam⁸⁵ (*TArgeted Reuse and GEneration of TEaching Materials*) é uma linguagem para apoiar a produção e gerenciamento de material de aprendizagem incluindo notas e conteúdo com explicações, motivações e exemplos.
- TML⁸⁶ (*Tutorial Mark-up Language*) é uma linguagem para produzir questões. Projetada para separar o conteúdo semântico da questão da sua forma de apresentação.
- IMS LD⁸⁷ é uma meta-linguagem para descrição de todos os elementos do projeto de um processo de ensino e aprendizagem. Serve para a definição de unidades de aprendizagem, ou seja, unidades elementares para prover eventos de aprendizagem, satisfazendo um ou mais objetivos de aprendizagem (TATTERSAL; KOPER, 2003).

Após a aprovação como uma especificação da IMS em fevereiro de 2003, a EML não é mais mantida ou atualizada e as atenções dos pesquisadores da OUNL estão agora direcionadas ao desenvolvimento da IMS LD.

O IMS LD permite o uso de diferentes abordagens de ensino aprendizagem (behavioristas, cognitivistas e construtivistas), em termos da definição dos papéis, atividades e recursos educativos. O resultado é um documento XML que pode ser processado por uma aplicação “*player*”, que coordena as interações dos envolvidos (estudantes, professores, monitores, etc.) com os materiais educacionais através da Web.

A especificação do IMS LD é baseada nos modelos conceitual, de informação e comportamental. O modelo conceitual traz a definição do vocabulário e as relações funcionais entre os conceitos do projeto de aprendizagem. O modelo de informação tem a descrição informal - linguagem natural - da semântica de cada conceito e

⁸³ <http://www.lmml.de>

⁸⁴ <http://sensei.ieec.uned.es/palo/>

⁸⁵ <http://www.targeteam.net>

⁸⁶ <http://www.ilt.bris.ac.uk/netquest>

⁸⁷ <http://www.imsglobal.org/learningdesign>

relação presente no modelo conceitual. O modelo comportamental apresenta a especificação das restrições presentes nos sistemas de software quando um dado projeto instrucional é executado.

No modelo conceitual observam-se os principais elementos globais: ambiente (*environment*), papel (*role*), atividade (*activity*) e o método (*method*), mostrados no diagrama de classes da UML na Figura 20. Os OA (*learning object*) compõem o ambiente (associação de agregação) e podem ser compostos por outros OA. Os papéis são generalizados como estudantes (*learner*) e administrativos (*staff*) - professores, monitores, agentes pedagógicos etc. As atividades são de aprendizagem (*learning activity*) ou de suporte (*support activity*) – monitoria, correção de avaliações, entre outras. Toda atividade faz parte de uma estrutura de atividades (*activity-structure*).

Todas as atividades estão associadas com um método onde os objetivos de aprendizagem (*learning objective*) e pré-requisitos (*prerequisite*) são conhecidos.

O modelo conceitual apresenta níveis semânticos, ou seja, diferentes níveis para sua implementação. O nível A é o núcleo principal, contém a diversidade pedagógica e os elementos globais: papéis, atividades, ambiente, método, objetivos, pré-requisitos, atividades de ensino e suporte. O nível B adiciona ao Nível A as propriedades (*property*), condições (*condition*) e elementos globais específicas como preferências, personalização, apresentação e mídias alternativas. O nível C acrescenta a estrutura de notificação (*notification*) ou mensagens acionada de acordo com o método, enviada para o ambiente e apoia um fluxo da aprendizagem.

O modelo conceitual traz também as propriedades de cada classe, ou seja, os metadados associados para elemento do projeto instrucional. A Figura D1 apresenta um fragmento do modelo com as propriedades para as classes *Learning-design*, *Components* e *Method*.

Figura D1 – Detalhes do Modelo Conceitual (IMS, 2007)

As principais etapas para construir uma unidade de aprendizagem são: definir a estratégia pedagógica, construir ou editar o projeto instrucional e executar o projeto instrucional.

Para a definição da estratégia pedagógica é preciso selecionar o domínio de conhecimento, o contexto, os objetivos, o público alvo, o modelo pedagógico desejado e o projeto instrucional.

Para construir o projeto instrucional é necessário selecionar os conteúdos e os serviços necessários. Os conteúdos englobam desde os OA até textos explicativos da atividade, tutoriais, perguntas e respostas frequentes. Os serviços são fundamentais para o Nível C pois a interação será maior. Assim serviços como de mensagens e *chats* são fundamentais.

Editar o projeto instrucional é dispor os conteúdos e serviços disponíveis em atividades e estruturá-los de acordo com o modelo pedagógico escolhido.

Após esta seleção e edição do modelo, faz-se necessário o empacotamento das unidades de aprendizagem para elas serem transferidas e interpretadas por LMS na entrega do projeto instrucional.

O Quadro D1 apresenta as ferramentas disponíveis para editar projetos de aprendizagem com o IMS LD.

Editor	Origem	Website	Nível
CopperAuthor	OUNL	www.copperauthor.org	A
Reload PA Editor	Reload	www.reload.ac.uk/ldeditor.html	A,B,C
ASK LDT	Universidade de Piraeus	www.ask.itl.gr	A,B
Mot+	Universidade de Quebec	www.liceftelug.quebec.ca/eng/productions/mot.html	A
Cosmos	Universidade de Duisburg	www.unfold-project.net:8085/UNFOLD/general_resources_folder/cosmos_tool.zip	A,B
Collage	Universidade de Valladolid	gsic.tel.uva.es/collage	A
LAMS	Universidade Macquarie	www.lamsinternational.com	A
Elive LD Suíte	Elive	www.elive-ld.com/content/index_ger.html	A, B

Quadro D1 - Ferramentas para edição IMS LD

Associadas com IMS LD estão várias empresas e organizações desenvolvendo ferramentas para sua edição, criação e execução.

Os requisitos para os editores são:

- ☐ completude: descrever o processo de ensino-aprendizagem em uma unidade de aprendizagem, incluindo referências para objetos de aprendizagem digitais ou não, e serviços necessários durante o processo.
- ☐ personalização: descrever os aspectos de personalização com o projeto de aprendizagem, ou seja, como as atividades e conteúdo das unidades de aprendizagem podem ser adaptados às preferências, necessidades educacionais, aos processos educacionais e às circunstâncias situacionais do usuário.
- ☐ compatibilidade: usar os padrões e especificações disponíveis tanto quanto possível.
- ☐ reuso: identificar, isolar, descontextualizar e trocar artefatos de aprendizagem importantes e reusá-los em outros contextos.
- ☐ facilidade de uso: combinar a visão gráfica em árvore com a representação gráfica da descrição dos relacionamentos.

Este padrão está sendo utilizado em alguns cenários pedagógicos como ensino colaborativo e ativo, ensino baseado em problemas, ensino por jogos, aprendizagem adaptativa, entre outros.

Assim surgem inúmeras possibilidades de representação para uma mesma unidade de aprendizagem como a personalização e apresentação alternativa de atividades ou OA de acordo com perfil do estudante ou do monitor, o acompanhamento da

execução em tempo real, formas alternativas de avaliação, acompanhamento da evolução do estudante, entre outras.

O IMS LD permite criar tanto seqüências simples e fixas como seqüências dependentes do contexto com retornos e interações com o estudante. Isto caracteriza os níveis possíveis de compatibilidade (A, B ou C) das ferramentas.

Exemplos de utilização e divulgação deste padrão são:

- Projeto IDLD⁸⁸ (*Implementation and Deployment of the Learning Design*) tem um portal com ferramentas e dicas metodológicas para construir projetos de aprendizagem de acordo com a especificação IMS LD. E possui também um repositório de projetos de aprendizagem (PAQUETTE, 2007).
- LN4LD⁸⁹ (*Learning Network for Learning Design*) é uma plataforma para aprender e trocar informações sobre a especificação IMS LD e sua aplicação. Apresenta unidades de aprendizagem e fóruns de discussão sobre vários aspectos da especificação.

⁸⁸ <http://www.idld.org>

⁸⁹ <http://imslld.learningnetworks.org/>

ANEXO E – Curso Computação Gráfica

O Interlab é um laboratório focado em pesquisa e desenvolvimento de novas tecnologias interativas como realidade aumentada, realidade virtual, hipermídia e jogos eletrônicos e suas aplicações em áreas como Educação, Engenharia e Saúde. Entre os principais projetos do Interlab se destacam ferramenta didática para ensino de conceitos de computação gráfica, ferramenta didática para projeto de jogos computacionais e infra-estrutura para realidade aumentada. No ensino de computação gráfica tem desenvolvido novas metodologias e ferramentas e aplicado em disciplinas presenciais de graduação na Escola Politécnica da USP no Departamento de Engenharia de Computação e Sistemas Digitais (PCS). A metodologia é baseada numa abordagem “*top-down*” e ferramentas de software livre, como Java 3D, *Art of Illusion* e Interlab 3D (desenvolvida para aprendizagem interativa) (TORI; NAKAMURA; BERNARDES, 2006).

Desde 2007 o Interlab participa da fase II do projeto Tidia-Ae como um Laboratório de Desenvolvimento Associado (LDA) realizando investigações e desenvolvimento de ferramentas relacionadas aos ambientes virtuais 3D, vídeo Avatar e Internet 2 bem como desenvolvimento de ações de aprendizagem. Neste contexto foi criada uma atividade de aprendizagem experimental a distância para o curso de Computação Gráfica existente, baseada na plataforma TIDIA-Ae, e seguindo as boas práticas sugeridas pelo IMS LD (BERNARDES et al; 2008). O projeto Tidia-Ae participa como membro votante do *IMS – Global Learning Consortium* – organismo responsável pela coordenação do padrão IMS LD. Três grandes etapas foram realizadas: definição, criação e edição da unidade de aprendizagem.

A. Definição da Unidade de Aprendizagem

O estudo das inúmeras possibilidades do IMS LD para reproduzir um cenário educacional numa UoL levou à realização das seguintes atividades:

1. Identificar os OA existentes. O curso de computação gráfica presencial existente possui diversos OA como apresentações para as aulas teóricas, definições de

exercícios, sugestões para projetos a serem desenvolvidos, avaliações, imagens e animações de vários conceitos apresentados. Nesta atividade foram discutidos aspectos como granularidade dos OA - um objeto de aprendizagem pode ser uma simples imagem ou um conjunto de recursos educacionais empacotados -, direitos autorais principalmente das imagens utilizadas nas apresentações e uso de algum LOR. Para a primeira versão, a granularidade adotada foi por formato do arquivo do objeto de aprendizagem como ppt, doc, pdf, gif entre outros, as imagens com direitos autorais devem ser substituídas ou obtidas sua permissão de uso e não será utilizada nenhuma ferramenta para LOR.

2. Identificar novos OA. Como as atividades podem ser realizadas a distância alguns novos OA são necessários como exercícios básicos e avançados, respostas dos exercícios, sugestões para projetos, projetos modelos, tutoriais, entre outros.
3. Selecionar o padrão de metadados para OA e o conjunto de atributos obrigatórios de preenchimento. O padrão de metadados adotado foi o IEEE LOM e como seus atributos são numerosos e opcionais foi adotado o conjunto de atributos recomendados do *United Kingdom Learning Object Metadata* (UK LOM)⁹⁰ bem como o domínio dos conteúdos sugeridos para cada atributo como para a linguagem utilizada (ISO 639:1988 e ISO 3166-1:1997).
4. Decidir o nível do IMS LD a ser considerado. O objetivo é criar um cenário educacional simples para servir como material de apoio para cursos de computação gráfica assim a primeira versão considera o nível A, ou seja, somente seqüência de atividades, papéis e ambiente. A utilização de perfis de papéis como estudantes novatos ou experientes e condições (nível B) ou de notificações e eventos combinados (nível C) ficará para as próximas versões.
5. Verificar o contexto cultural como idioma e exemplos utilizados. Para a primeira versão o idioma escolhido foi o português e os exemplos utilizados não apresentam características comprometedoras referentes aos aspectos culturais como política ou religião.

⁹⁰ <http://zope.cetis.ac.uk/profiles/uklomcore>

6. Identificar tutoriais necessários de acordo com os pré-requisitos e/ou inabilidades do estudante. Vários tutoriais já existem principalmente para uso das ferramentas utilizadas nas aulas práticas como Interlab3D (INTERLAB, 2008) e EnJine (NAKAMURA; BERNARDES; TORI, 2006).
7. Identificar os papéis presentes na UoL. Para a primeira versão foram identificados os papéis aluno, professor de teoria e professor de laboratório.
8. Identificar os serviços necessários como fórum, mensagens ou laboratório virtual. Para a primeira versão são necessários serviços para visualização dos OA e para *download* das ferramentas necessárias para construção do ambiente prático pelo aluno.
9. Identificar os materiais de apoio necessários como FAQ, glossário, referências, documentos extras entre outros. O ideal para o ensino a distância é promover todos os recursos para facilitar a execução das atividades definidas e a exploração dos OA disponibilizados.

B. Criação da Unidade de Aprendizagem

Com as definições acordadas, discussões realizadas e recursos educacionais selecionados foram realizadas duas importantes atividades para criação da UoL: construir a narrativa do cenário didático e expressar a narrativa num diagrama de atividades da UML.

1. Construir a narrativa do cenário didático.

A narrativa partiu da análise dos recursos educacionais existentes como ementa da disciplina, artigos publicados - Ferreira e Tori (1998), Tori e Ferreira (1998) e Tori, Nakamura e Bernardes (2006) entre outros -, e discussões com os professores responsáveis pela disciplina.

Para orientar e documentar a narrativa foi adaptado o modelo sugerido no Projeto IDLD no estudo de caso TÉLUQ (PAQUETTE, 2007). O resultado pode ser visto no Quadro E1.

Análise das Tarefas	Descrição
Título	Computação Gráfica
Descrição Principal e abordagem pedagógica	Abordagem <i>top-down</i> onde os alunos manipulam estruturas 3D inicialmente nas aulas de laboratório e estudam com maior profundidade nas aulas teóricas posteriormente. Atividades individuais e em grupo.
Tempo previsto	15 semanas curso (aproximadamente 60 horas na sala de aula somadas a 40 horas no ambiente prático).
Contexto	A disciplina exige a definição e construção de um jogo. Usando um motor de jogo os estudantes são capazes de construir jogos complexos e atraentes com o uso dos principais conceitos da CG.
Pré-requisitos	Bons conhecimentos de programação Java e orientação a objeto.
Objetivos de aprendizagem	Explorar o desenvolvimento de algoritmos e o uso de ferramentas para o desenvolvimento de sistemas gráficos. Introduzir os fundamentos da Computação Gráfica e Processamento de imagens através da programação de sistemas, utilizando bibliotecas gráficas e sistemas de realidade virtual. Uso de sistemas gráficos de visualização tridimensional.
Papéis	Professor Teoria e Professor do Laboratório Alunos e grupos de trabalho
Serviços	Portal do Projeto Tidia-Ae <ul style="list-style-type: none"> • Email (SM) • Índice de todas atividades previstas • Índice de todos OA

Quadro E1 – Narrativa do Curso de Computação Gráfica – Parte I

Atos	<p>O curso está dividido em duas grandes atividades: Teoria e Prática em três grandes módulos.</p> <p>Módulo I - Teoria</p> <ul style="list-style-type: none"> • Introdução ao curso e CG: origem, classificação e dispositivos; • Ótica, visão humana e sistema de cor; • Tarefa: proposta de um jogo. <p>Módulo I - Prática</p> <ul style="list-style-type: none"> • Uso de uma ferramenta para modelar 3D; • Conceitos: modelo sólido, curvas e superfícies, polígonos, cores, transparência, texturas, luminosidade, sombreamento, entre outros. • Tarefa: modelar componentes estáticos do jogo <p>Módulo II - Teoria</p> <ul style="list-style-type: none"> • Representação e modelagem de cenários e sólidos; • Gráficos; • Iluminação local e global; textura, iluminação entre outros; <p>Módulo II - Prática</p> <ul style="list-style-type: none"> • Introdução aos cenários gráficos • Conceitos: animação, transformações geométricas, cenários gráficos • Tarefa: construir os componentes dinâmicos do jogo <p>Módulo III - Teoria</p> <ul style="list-style-type: none"> • Transformações geométricas, primitivas gráficas e projeções; • Curvas e superfícies; • Modelagem e Visualização em 3D; • Sistemas gráficos padrões; • Tópicos avançados como realismo, animação, visualização, VR, AR, entre outros. <p>Módulo III - Prática</p> <ul style="list-style-type: none"> • Programação do jogo com um motor • Conceitos: superposição 2D, manipulação câmera, interação com usuário, entre outros • Tarefa: Jogo construído.
Estrutura das Atividades	<p>Estrutura 1: Teoria</p> <p>Estrutura 2: Prática</p>

Quadro E1 – Narrativa do Curso de Computação Gráfica – Parte II

2. Expressar a narrativa num diagrama de atividades da UML.

As atividades identificadas na narrativa foram traduzidas e detalhadas num diagrama de atividades da UML ressaltando os papéis (quem executa) – professor de teoria, aluno e professor de laboratório -, os OA participantes da atividade (o que usar), a sequência, paralelismo e dependência das atividades, outros recursos necessários (descrição, objetivos esperados, tutoriais, recursos externos necessários como URLs entre outros) e os resultados esperados (produtos finais resultantes).

Nesta modelagem foram seguidos padrões para as nomenclaturas utilizadas no diagrama de acordo com os quadros: Quadro E2 para os verbos utilizados na definição das atividades do aluno, Quadro E3 para os verbos utilizados na definição das atividades dos professores e o Quadro E4 para identificar os recursos de entrada e saída das atividades.

Verbo	Descrição
Ler	Leitura de algum documento ou tutorial.
Analisar	Análise de algum projeto ou artigo.
Programar	Programação de atividades ou geração de componentes para o projeto final.
Integrar	Integração de componentes no sistema construído.
Responder	Resposta a alguma avaliação ou questão.
Projetar	Elaboração do projeto de um sistema.
Praticar	Por em prática algum método ou ferramenta.
Publicar	Publica algo na Web.
Explorar	Verificar todos recursos oferecidos pelos OA.

Quadro E2 – Verbos para atividades do aluno

Verbo	Descrição
Apresentar	Exposição de conteúdo numa apresentação. Apresentações serão entregues aos alunos naturalmente.
Fornecer	Entrega de algum documento para o aluno.
Supervisionar	Assistência ao aluno em alguma atividade.
Avaliar	Avaliação do progresso do aluno.

Quadro E3 – Verbos para atividades dos professores

Sigla	Descrição
DOC	Um documento texto que pode ser um documento oficial elaborado pela instituição, ou informal como um preparado por um professor.
LO	Objeto de aprendizagem - pode ser um texto (TX), um tutorial (TU), uma apresentação (PPT) ou outro tipo de mídia como animação (AN), vídeo (VD), simulação (SI) etc.
PRG	Programa.
EV	Avaliação; exame.
EX	Exercícios.

Quadro E4 – Siglas para os recursos de entrada e saída das atividades

O diagrama de seqüência resultante da narrativa do Módulo II para o tópico “Iluminação” e dos padrões adotados pode ser visto na Figura E1.

Figura E1 –Diagrama de Atividades do Módulo II do CCG

As atividades do lado esquerdo do diagrama correspondem às atividades teóricas e as do lado direito às atividades práticas. O aluno pode explorar dois OA (LO10 e LO11) sobre iluminação, depois outro sobre mapeamento de texturas (LO12). Paralelamente pode ler um tutorial (TU4) do Interlab 3D, realizar o exercício prático

em Java 3D (EX2) e programar (EX3) componentes dinâmicos do jogo (PRG2) podendo consultar os componentes dinâmicos (EX3). Após estas atividades o aluno realiza uma primeira avaliação (EV7).

Uma lista dos OA participantes com sua identificação, subtipo, tipo, nome e arquivo físico correspondente foi elaborada e permitiu destacar os OA necessários de serem construídos.

C. Edição da Unidade de Aprendizagem

Parte do diagrama de atividades elaborado foi editado num sistema educacional para desenvolvimento e autoria de recursos educacionais aderente ao IMS LD, ou seja, na ferramenta Reload Editor. Esta etapa contou com as seguintes atividades:

1. Aprender os recursos da ferramenta. A ferramenta está estruturada de acordo com grandes tópicos abordados no IMS LD: descrição da UoL com objetivos e pré-requisitos (*overview*), papéis (*roles*), propriedades (*properties*), atividades (*activities*), ambiente (*environments*) e método (*method*). Além disso, oferece outras opções como arquivos existentes (*files*) e geração do arquivo compactado (*export*) com todos os arquivos referenciados, metadados e documento XML correspondente chamado *imsmanifest*. A estrutura de diretórios é criada a partir do nome da UoL seguida por outros três diretórios: *content*, *metadata* e *output*. No primeiro ficam os documentos e arquivos existentes para a UoL, no segundo os metadados existentes e no terceiro os arquivos referenciados na UoL e o IMS *Manifest* correspondente. Para facilitar o uso foi criado um tutorial da ferramenta.
2. Criar documentos relativos às atividades. A ferramenta permite criar documentos específicos para descrição, objetivos e pré-requisitos de cada atividade e de seus respectivos metadados. Estes documentos podem ser em diferentes formatos e múltiplos para mesma atividade. Para primeira versão foram construídos arquivos textos com a descrição da atividade.
3. Criar os metadados dos OA. A criação dos metadados dos OA é uma tarefa trabalhosa e repetitiva, pois muitos atributos do metadado são os mesmos para um mesmo ambiente. Para melhorar isto, duas ações foram realizadas: criação do

perfil Interlab e definição do utilitário para extração parcial dos metadados. A ferramenta Reload permite criar perfis (*profiles*) padrão para uso na criação dos metadados e então foi criado o “*profile* Interlab”, baseado no UK LOM, onde além da obrigatoriedade ou não dos atributos, valores iniciais (*default*) foram sugeridos. Assim cada novo metadado inserido ou alterado manualmente seguirá este perfil. Para facilitar a definição dos metadados para os OA existentes – arquivos presentes no diretório *content* - foi definido um utilitário para extrair as características dos arquivos como nome (*title*), formato (*Technical – Format*), data de criação (*Life Cycle*), tamanho (*Technical – Size*) e localização (*Technical – Location*), para adicionar os valores iniciais definidos no *profile* Interlab e para gerar parte do metadado automaticamente. Estas ações não eliminaram a necessidade de correção e inserção manual de todos atributos importantes para o metadado de cada OA como as palavras-chaves.

4. Editar as atividades. A transposição incremental do diagrama de atividades na ferramenta foi realizada.
5. Exportar atividades. O arquivo exportado no formato IMS-LD nível A descreve um conjunto de estruturas padronizadas para prover interoperabilidade de conteúdo entre ferramentas de autoria e ambientes de publicação e entrega, ou seja, um pacote - arquivo compactado - que pode ser interpretado por diversos LMS. Este pacote contém um arquivo IMS *Manifest* (*imsmanifest.xml*) e os arquivos referenciados por ele (DTD, XSD e recursos utilizados).
6. Testar a execução da UoL. Para verificar a validade da representação obtida e para visualizar a execução da UoL foi utilizada a ferramenta *Reload Learning Design Player*. Qualquer inconsistência existente nos arquivos de saída (*output*) principalmente no *imsmanifest.xml*, provoca erros na execução e demanda alterações na edição. Vale destacar que esta ferramenta não está integrada a nenhum LMS. A Figura E2 mostra a apresentação gerada para o aluno onde no primeiro quadro da esquerda estão as atividades, no segundo o ambiente com os OA disponíveis e no quadro da direita uma descrição sobre a atividade.

Figura E2 – Módulo II do CCG no Reload Player

A etapa para execução da UoL no ambiente Tidia-Ae não foi contemplada neste trabalho porém estudos preliminares mostram sua viabilidade de implementação como o modelo conceitual existente (BRESSAN; RUGGIERO, 2005), as principais características e funcionalidades necessárias (BOARETTO; NUNES; FILATRO, 2006) e a compatibilidade com a ferramenta Sakai⁹¹.

⁹¹ <http://confluence.sakaiproject.org/confluence/display/MJN/Learning+Design+Project>