Combining Methontology and Ontology Driven Approach to Build an Educational Ontology

Marlos Silva, Endhe Elias, Evandro Costa, Ig Ibert Bittencourt, Heitor Barros, Leandro Dias da Silva, Alan Pedro da Silva, Douglas Véras

Abstract— **Intelligent Learning Environments can be broadly defined as computer-based educational systems that rely on diverse Artificial Intelligence techniques to improve students learning experience, and help them reach their learning objectives. In this context, when it comes to formal and sharable representation of knowledge, Semantic Web technologies in general, and ontologies in particular offer a promising solution to represent knowledge. On the other hand, ontologies are more closely related to concepts of Artificial Intelligence paradigms and are typically unknown to a large population of software engineers. Still obvious overlaps between both fields is emerging a hybrid approach to systems development, combining Semantic Web technologies and techniques with more established development formalisms and languages like the Unified Modeling Language. Therefore, this paper proposes to combine the Methontology and a Model Driven Architecture approach to build educational ontologies.**

Index Terms — ***Education, Ontology, Ontology Engineering, Model Driven Approach.***

I. Introduction

Intelligent Learning Environments (ILES) can be defined as computer-based educational systems that rely on Artificial Intelligence (AI) techniques to improve students' learning experience, and help them reach their learning objectives. The primary advantage of ILEs over more traditional learning systems and tools lays in their ability to provide students with individualized interactions tailored to a student's personal traits (e.g., goals, tasks, knowledge, experiences, preferences, and individual traits), as well as the characteristics of the student's environment (e.g., location, computing platform, and bandwidth) [1].

In this context, modern ILES have tried to incorporate Semantic Web resources into their designs and architectures. The main idea behind this is the attempt of both representing information on the Web in a way that computers can understand and manipulate it and providing the learning environments more adaptable, personalized, and intelligent [2].

On the other hand, although the ontologies present a series of improvements in representation of knowledge, they are more closely related to concepts of Artificial Intelligence paradigms (e.g., Description Logics, Semantic Networks, and Frames). Accordingly, most of the current Semantic Web ontologies have been developed in Artificial Intelligence laboratories [3]. Moreover, the Semantic Web technologies should not be a substitute for current practices, rather a complement to them, because software engineers will not abandon their experiences.

Until recently work on accepted practices in Systems and Software Engineering has appeared somewhat disjointed from that breaking ground in the area of formal information representation on the World Wide Web (commonly referred to as the Semantic Web initiative[1]. Still obvious overlaps between both fields are apparent and many now acknowledge merit in a hybrid approach to systems development, combining Semantic Web technologies and techniques with more established development formalisms and languages like the Unified Modeling Language (UML) [4]. This is not only for the betterment of IT systems in general, but also for the future good of the Web.

Over the time a few proposals have been made suggesting the use of software engineering techniques, especially the UML since it is the most accepted software engineering standard [5]. Therefore, some authors propose the use of software UML extensions [i.e., UML profiles] and OMG's Model Driven Architecture methodology, for ontology development [6] [7] [8]. Then, the software engineers can be use models, based on UML extensions, in order to define their ontologies.

This paper presents a combination of Methontology, which is an ontology methodology, and a Model Driven approach to specify educational ontologies based on the Mathema model [9]. In the specification process is used the Methontology, in the formalization process is used the Ontology UML Profile (OUP) [3], that is based on Ontology Web Language (OWL) [10], and a XSLT [11] approach is used for the transformation in OWL ontologies.

The paper is organized as follow. In Section II an introduction to the concepts related to this work is presented. In Section III the conceptualization and implementation of educational ontologies is described. In Section IV a discussion about some aspects of this work is presented. In Section V is presented the conclusions of the paper and is discussed the further works.

II. Background

This section describes the background of this work, and presents the definitions of Mathema Model. Furthermore, is

Manuscript received January 20, 2011.

Publisher Identification Number 1558-7908-2011-13

[1] http://www.w3.org/2001/sw/

presented METHONTOLOGY, a methodology to build ontologies, and finally is described a Model Driven Approach and its applications in the ontology development.

A. The Mathema Model

In the Mathema conceptual model [12] is proposed an architecture to Intelligent Tutoring Systems that consists of three modules: i) the Society of Tutoring Agents (TAS); ii) the apprentice interface; and iii) the authoring interface. The society of tutoring agents consists of a multi-agent system where each agent contains a complete intelligent tutoring system focused on a sub-domain of the target domain.

Each one of the intelligent tutoring agents in the TAS is responsible by one sub-domain. The Mathema conceptual model [9] provides a modeling scheme for these sub-domains that is divided into two views: the external view; and the internal view.

The external view is a partitioning scheme based on epistemological assumptions about the domain knowledge and consists of a three dimensional perspective [12] [13]:

- **Context:** a knowledge of domain might be composed by different contexts or points of view. In other words, a knowledge object might be several interpretations or might be represented by different approaches in the same domain knowledge;
- **Depth:** given one particular context, the associated knowledge might be partitioned according to the methodologies used to deal with its contents;
- **Laterality:** given one particular context and one particular depth, complementary knowledge might be pointed to in order to allow the apprentice to acquire background knowledge not in the scope of the course or to reach related additional contents.

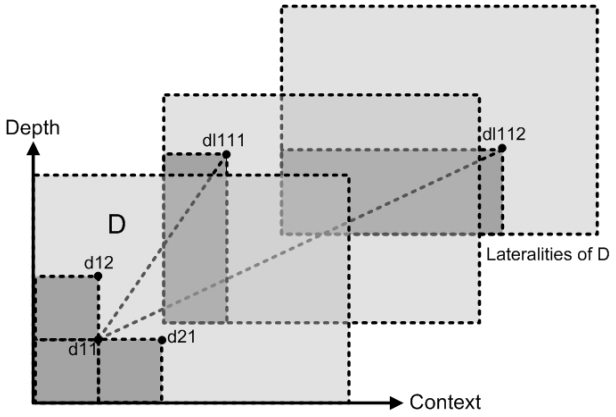

Fig. 1. Multidimensional View of Domain Knowledge (adapted from [9])

Therefore, with this multidimensional approach (Figure 1) is possible to focus a domain by a contextual view, and this view might be accompanied by several alternatives of depth and laterality, each one representing a different methodological approach.

Once the external view is constructed, the internal view is applied in each sub-domain. In the process of construction of the internal view, the knowledge associated with each subdomain has to be organized into a set of curricula. Each curriculum is progressively refined according to three levels:

- **First Level:** in the first level of the internal view, each curriculum is detailed into a set of pedagogical units. Theses pedagogical units are used to describe a possible sequence of contents which are presented to the apprentice, and might be ordered based on educational criteria;
- **Second Level:** in the second level, each pedagogical unit is refined into a set of problems, also ordered and possible with prerequisites relationships;
- **Third Level:** in the third level, each problem is associated with a set of interaction units with the apprentice, that support the problem solving activities, such as explanations, examples and exercises.

The pedagogical structure of the domain has three fundamental plans which are denominated: pedagogical plan; plan problems; and support plan. As can be seen in Figure 2, the pedagogical plan (higher plan) has the pedagogical units that are related by prerequisites and levels of difficulty, aiming to indicate an order of execution of educational activities. In the plan problems (mid plane) are sets of problems, and each of them is associated with a pedagogical unit, serving as a basic pedagogical activity in the teaching-learning process. These problems are also related by a partial order defined in accordance with the level of difficulty involved in their resolution. Finally, in the support plan (lower plan) is defined the knowledge use to support the resolution of the problems. Thus, is included: concepts, examples, hints, similar problems, and among others.

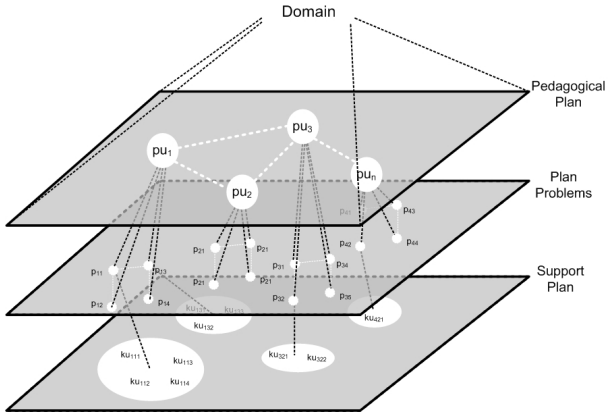

Fig. 2. Pedagogical Structure of Domain Knowledge (adapted from [9])

B. Methontology

Methontology was developed within the Ontology Group at Universidad Politécnica de Madrid. This methodology [14][15][16] has been proposed for ontology construction by the Foundation of Intelligent Physical Agents (FIPA[2]), which promotes inter-operability across agent-based applications. It proposes an ontology building life cycle based on evolving

[2] http://www.fipa.org/

prototypes because it allows adding, changing, and removing terms in each new version, as can be seen in Figure 3.

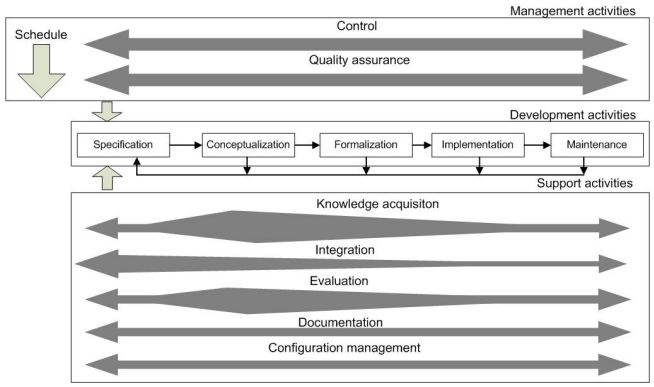

Fig. 3. Development life cycle of Methontology (adapted from [17])

Once the specification model is built, the conceptualization activity in Methontology organizes and converts an informally perceived view of a domain into a semi-formal specification using a set of intermediate representations based on tabular and graph notations that can be understood by domain experts and ontology developers. After the conceptual model was built, the methodology proposes to transform the conceptual model into a formalized model (e.g., Description Logic [18], Ontology UML Profile [3], among others), which will be implemented in an ontology implementation language (e.g., Ontology Web Language [10]). That is, along this process the knowledge engineering is moving gradually from the knowledge level to the implementation level, increasing slowly the degree of formality of the knowledge model so that it can be understood by a machine. Methontology focus on the conceptualization models and defines a set of tasks that have to be followed by the knowledge engineer. Each task is described in the following [17].

- **Task 1:** To build the glossary of terms that identifies the set of terms to be included on the ontology, their natural language definition, and their synonyms and acronyms;
- **Task 2:** To build concept taxonomies to classify concepts. The output of this task could be one or more taxonomies where concepts are classified;
- **Task 3:** To build ad hoc binary relation diagrams to identify ad hoc relationships between concepts of the ontology and with concepts of other ontologies;
- **Task 4:** To build the concept dictionary, which mainly includes the concept instances6 for each concept, their instance and class attributes, and their ad hoc relations;
- **Task 5:** To describe in detail each ad hoc binary relation that appears on the ad hoc binary relation diagram and on the concept dictionary. The result of this task is the ad hoc binary relation table;
- **Task 6:** To describe in detail each instance attribute that appears on the concept dictionary. The result of this task is the table where instance attributes are described;
- **Task 7:** To describe in detail each class attribute that appears on the concept dictionary. The result of this task is the table where class attributes are described;
- **Task 8:** To describe in detail each constant and to produce a constant table. Constants specify information related to the domain of knowledge, they always take the same value, and are normally used in formulas.

The knowledge engineer should describe formal axioms (task 9) and rules (task 10) that are used for constraint checking and for inferring values for attributes. And only optionally should the knowledge engineer introduce information about instances (task 11). Once the conceptual model is built, the methodology proposes to transform the conceptual model into a formalized model (e.g., Description Logic Model, F-Logic Model, ontology UML profile, among others), which will be implemented in an ontology implementation language (e.g., Ontology Web Language, among others).

C. *Model Driven Approach*

Over the years, MDA approach has provided a good support and consolidation for the automatic generation of code for application development [19]. Moreover, several companies apply MDA, for example, IBM, and it is positive approach because: encourages efficient use of system models in the software development process, and it supports reuse of best practices when creating families of systems [20].

The Object Management Group (OMG) defines that MDA has goal isolate business logic from the application, development and maintenance of technology. Thus, using this approach, the systems can be built quickly, consistently and platform independent[3]. Three level of abstraction compose MDA: Computation Independent Model (CIM), Platform Independent Model (PIM) e Platform Specific Model (PSM).

The first level is CIM, it is a view of a system that does not show the details of a system structure, it is also known in the literature by the domain model, similar to the same concept presented in ontology [21]. On the other hand, the PIM also has high level of abstraction and it is independent of any technology implementation. Furthermore, all functional are defined in PIM, as well as, business constraints. Finally, the PSM is a model that will be associated in specified system. However, one PIM can be transform into various PSM. The transformations are done automatically, using tools. This process can be seen in Figure 4.

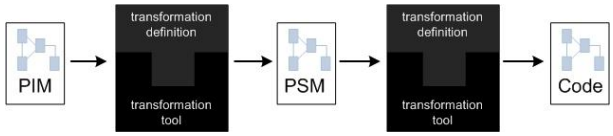

Fig. 4. Transformation Process between models (adapted from [22])

As can be seen, a model-driven approach has several advantages, among which stand out: changing requirements affect the most abstract model and this change reflected to the other models automatically, it encourages software reuse,

[3] http://www.omg.org/mda/

among others. This approach can be applied to the ontology engineering process, as will be seen below.

D. MDA applied to Ontology Engineering

There are several works in literature that show MDA applied to ontology engineering process, for example, in [23] is defined Ontology Driven Architectures (ODA), it combines MDA with the semantic technology differently from the ODM approach. ODA attempts to augment the MDA standards and methodology stack with the semantic technology to improve the discipline. It aims to enable unambiguous representation of domain terminology, distinct from the rules, enable automated consistency checking and validation of invariant rules, preconditions, and post-conditions, and support knowledge based terminology mediation and transformation for increased scalability and composition of components.

On the other hand, [21] proposes combine MDA and Ontology Definition Meta-model, which enable model transformation. This work allows seamlessly supporting existing models in UML and other languages in Semantic Web-based software development. In addition, it allows exploiting the availability and features of UML tools for creation of vocabularies and ontologies.

Although there is a difference between MDA and the process of ontology engineering, in [3] is presented several ways and their advantages that how MDA can be applied in process of ontology engineering and this kind of approach has been increasingly applied in the software industry.

III. THE EDUCATIONAL ONTOLOGY

In this section is presented the process of construction of the MATHEMA Model. This specification is based on the tasks defined by the METHONTOLOGY [16]. For the formalization of the model we use an Ontology UML Profile [24] that is based on OWL language [10]. For the implementation of the model we use an automatic transformation, based on a XSLT approach [11], of the model described in UML into OWL ontology.

A. Specifying the Ontology with Methontology

In the first task is built a glossary of terms that includes all relevant terms of the domain, their natural language description and, if necessary, their synonyms and acronyms. In Table I we illustrate all the relevant terms of the Mathema Model. However, is important to mention that on the initial stages of the ontology conceptualization the glossary of terms might has terms that refer to the same concept.

In the second task of the conceptualization activity is proposed to build concept taxonomies to define the concept hierarchy. To do this, the knowledge engineering selects terms that are concepts from the glossary of terms. Methontology proposes to use four taxonomic relations: Subclass-Of; Disjoint-Decomposition; Exhaustive-Decomposition; and Partition. In the Subclass-Of relationship, a concept C is a parent of another concept D, in other words, every instance of the concept C is also an instance of the concept D. In the Disjoint-Decomposition relationship, a concept C has a set of subclasses that do not have common instances and do not cover the superclass, in this case the concept C. In other words, there might be instances of the concept C that are not instances of any of the concepts in the decomposition. In the Exhaustive-Decomposition relationship, a concept C has a set of subclasses that cover the superclass and might have common instances. In other words, there cannot have instances of the concept C that are not instances of at least one of the concepts in the decomposition. In the Partition relationship, a concept C has a set of subclasses that do not share common instances but that cover the superclass. That is, there are not instances of the concept C that are not instances of only one of the concepts in the partition. However, the Mathema Model does not present a sizable number of terms and, thereby, this task was ignored.

In the third task the conceptualization activity proposes to define binary relation diagrams. The goal of this task is to establish relationships between concepts of the same (or different) concept taxonomy. The binary relation diagrams do not be presented in this paper, however all the relationships might be viewed the fifth task.

In the fourth task the conceptualization activity proposes to specify which are the properties and relations that describe each concept of the taxonomy. This concept dictionary (Table II) contains all the domain concepts, their relations, and their attributes. Relations and attributes are local concepts, in other words, it means that their names might be repeated in different concepts. Furthermore, attributes might be divided into two groups: the class attributes; and the instance attributes. The class attributes describe concepts and take their values in the class where they are defined. The instance attributes describe concepts and take their values in the instance where they are defined.

TABLE II
The Concept Dictionary of Mathema Model

Concept name	Instance attributes	Relations
Domain	name description	composed by Partition
Context	name description	composed by Depth
Depth	name description	
Curriculum	name description	composed by Context composed by Depth composed by Laterality composed by Pedagogical Unit
Pedagogical Unit	name description	composed by Problem has next Unit has previous Unit
Problem	name description solution	composed by Knowledge Unit
Knowledge Unit	name description	composed by Learning Objects

In the fifth task, is possible to detail the relations and produce a binary relation table. So, for each relation identified, the knowledge engineer has to specify its name, the names of the source and target concepts, its cardinality, its inverse relation and mathematical properties (e.g., symmetry, transitivity, reflexivity, among others). The Table III shows the binary relation table of the Mathema Model. However, the fields of inverse relation and mathematical properties were removed, because none of them were defined in this model.

TABLE I
The Glossary of Terms of Mathema Model

Name	Description	Type
Domain	Knowledge Domain	Concept
Context	Different points of view about a knowledge domain	Concept
Depth	Some kind of sophistication of a given context	Concept
Laterality	Prerequisites which are related to others knowledge domains	Concept
Curriculum	Partitions of a given knowledge domain	Concept
Pedagogical Unit	Units that define the sequence in which problems are presented	Concept
Problem	Some kind of problem to be solved	Concept
Knowledge Unit	Knowledge support necessary for resolution of the problem	Concept
Learning Object	Some kind of learning object	Concept
name	The name of domains, contexts, depths, and etc.	Attribute
description	The description of domains, contexts, depths, and etc.	Attribute
solution	The solution of a problem	Attribute
composed by Partition	The partitions in which the domain might be divided	Relation
composed by Context	The points of view in which the domain might be approached	Relation
composed by Depth	The sophistication related to a given context	Relation
composed by Laterality	The prerequisites related to a given curriculum	Relation
composed by Pedagogical Unit	The pedagogical units related to a given curriculum	Relation
composed by Problem	The problems related to a given pedagogical unit	Relation
has next Unit	The next pedagogical unit	Relation
has previous Unit	The previous pedagogical unit	Relation
composed by Knowledge Unit	The knowledge units related to a given problem	Relation
composed by Learning Objects	The learning objects related to a given knowledge unit	Relation

In the sixth task is described all the instance attributes. For each instance attribute, the knowledge engineer has to detail the information with the following fields: its name; the concept it belongs to; its value type; its measurement unit, precision and range of values (in the case of numerical values); default values if they exist; minimum and maximum cardinality. The Table IV shows the instance attribute table of the Mathema Model. However, the fields of measurement unit, precision, range values and default values were removed, because none of them were defined in this model.

TABLE IV
Instance Attributes os Mathema Model

Concept name	Attribute Name	Value Type	Cardinality
Domain	name	String	(1,1)
	description	String	(0,1)
Context	name	String	(1,1)
	description	String	(0,1)
Depth	name	String	(1,1)
	description	String	(0,1)
Curriculum	name	String	(1,1)
	description	String	(0,1)
Pedagogical Unit	name	String	(1,1)
	description	String	(0,1)
Problem	name	String	(1,1)
	description	String	(1,1)
	solution	String	(1,N)
Knowledge Unit	name	String	(1,1)
	description	String	(0,1)

In the seventh task is defined all the class attributes. For each instance attribute, the knowledge engineer has to detail the information with the following fields: name; the name of the concept where the attribute is defined; value type; measurement unit and value precision (in the case of numerical values); and cardinality. The eighth task is used to describe the constants defined into his glossary term. For each constant, the knowledge engineer has to define the following fields: name; value type (e.g., a number, a string, among others); value; and the measurement unit for numerical constants. In the ninth and tenth tasks are defined the axioms and rules respectively. The last four tasks were ignored, because they do not present any contribution to the specification of this model. However, is important to stress that these tasks are important and might be used to specify another kinds of models.

B. Formalizing the Ontology with UML

For the ontology formalization is used an Ontology UML (OUP) [3] that is an extension of UML which provides a set of stereotypes. Basically, the OUP defines the following stereotypes: the *<<OntClass>>* stereotype; the *<<DataProperty>>*; the *<<ObjectProperty>>* stereotype; and *<<range>>* and *<<domain>>* stereotypes. The *<<OntClass>>* stereotype is used to define the ontology classes, for example, Context, Depth, Curriculum, and among others. The *<<DataProperty>>* stereotype is used to define the attributes, for example, name, description, solution, and among others. The *<<ObjectProperty>>* stereotype is used to define the relations, for example, is compose by contexts, has previous unit, and among others. The *<<range>>* and *<<domain>>* stereotypes are used to interconnect the classes with the relationships and attributes, for example, in the relationship Context is composed by a Depth, the Context is the domain of the relation and the Depth is the range. Also, is possible to define the cardinality of the relationships.

Based on the specification of the Mathema Ontology, is possible to use the stereotypes of the Ontology UML Profile for formalizing the ontology. As can be seen in the Figure 5, each class defined in the Methontology specification is translated into a class with the *<<OntClass>>* stereotype. For each relationship is used the class with the *<<ObjectProperty>>* stereotype and the stereotypes *<<range>>* and *<<domain>>* are used to interconnect the classes.

TABLE III
The Binary Relation of Mathema Model

Source concept	Relation Name	Source Cardinality	Target concept
Domain	composed by Partition	N	Curriculum
Context	composed by Depth	N	Depth
Curriculum	composed by Context	1	Context
	composed by Depth	1	Depth
	composed by Laterality	N	Curriculum
	composed by Pedagogical Unit	N	Pedagogical Unit
Pedagogical Unit	has next Unit	1	Pedagogical Unit
	has previous Unit	1	Pedagogical Unit
	composed by Problem	N	Problem
Problem	composed by Knowledge Unit	N	Knowledge Unit
Knowledge Unit	composed by Learning Objects	N	Learning Object

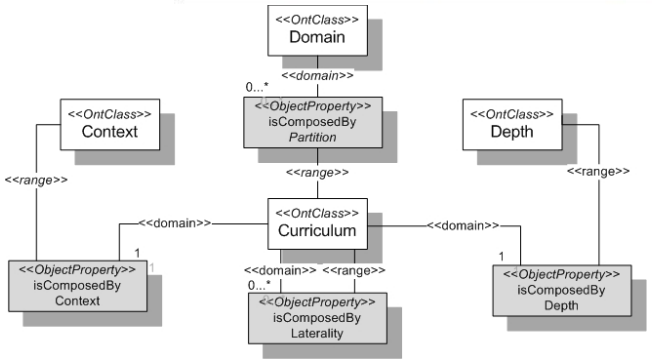

Fig. 5. External View of the Mathema Ontology

After the definition of models it is possible to generate the correspondent ontology in OWL language. A UML tool (e.g., Poseidon for UML, Magic Draw, among others) can export a XMI document, which a XSLT processor can use as input [11]. Therefore, an OWL document is produced as output and this format can be imported into a tool specialized for ontology development (e.g., Protégé[4]), where it can be further refined. The OWL-based ontology can be viewed in Figure 6.

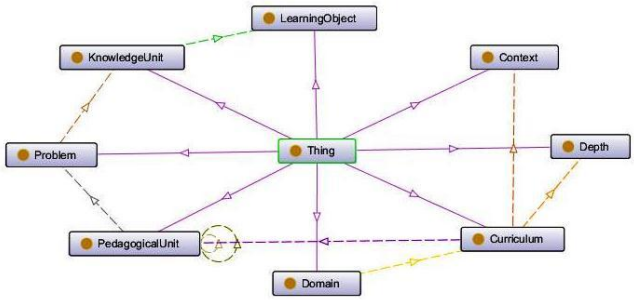

Fig. 6. Mathema Ontology in Protégé Tool

IV. Discussion

The specification of an educational system is costly because it involves a great range of concepts (e.g., domains, curricula, students, goals, among others). Also, the purely syntactic description of resources makes it harder to reuse, sharing, and automatic processing of information. However, the use of ontologies can further complicate this process because it involves a lot of other concepts (e.g. entities, relationships, rules, axioms, among others).

[4] http://protege.stanford.edu/

In this context, three issues might prevent the growth of the Semantic Web: a lack of simplicity, integration with existing technologies and practices, and adoption by the industry. These three issues can easily fit the area of Intelligent Learning Environments, which does not have a widely accepted approach. Therefore, this study presents a solution to these problems:

- **Simplicity:** the use of a Model Driven Architecture combined with the Methontology tasks reduces the necessary prerequisites. Thus, a software engineer can specify their own ontologies, because he uses a methodology to guide the ontology specification and high level models based on a standard (i.e., UML). As a result, the specification of educational models became the process of definition much more intuitive. Moreover, the Model Driven Architecture enables the automatic transformation between models and can automatically transform the modeled system into ontologies;
- **Integration:** this approach follows the standards of Model Driven Architecture. So, is possible to use a set of techniques and technologies consolidated for many years by Software Engineering professionals. Furthermore, there are currently a large number of tools that support UML modeling, and a great number of OMG's standards.
- **Industry acceptance:** this approach can be more easily absorbed by professionals of industry because did not drastically change the way the system is designed. The definition of a high level model for specifying educational ontologies unites professionals in education, software engineering and artificial intelligence areas. Furthermore, this approach combines the facilities provided by a Model Driven Approach with the advantages of the Ontology Engineering.

V. Conclusions

The main idea of this paper is to define an ontology combining the Methontology and the Model Driven Architecture, such as Ontology UML profile. Therefore, this approach provides not only an educational model that can be more easily shared, reused, extended and processed but also uses a set of consolidated ontology engineering and software engineering techniques, for ontology specification. Moreover, this approach is in conformance with Semantic Web

Infrastructure and Semantic Web Languages (e.g., RDF, and OWL).

As a future work, we will define a knowledge tool in order to help in the process of the ontology specification, formalization and implementation. Furthermore, it will be in conformance with OWL 2.0.

REFERENCES

[1] J. Jovanovíc, D. Gasevic, C. Torniaic, S. Batemand, and M. Hatala, "The social semantic web in intelligent learning environments: state of the art and future challenges," *Interactive Learning Environments*, vol. 2, pp. 273 – 309, 2009.

[2] I. I. Bittencourt, E. de Barros Costa, M. Silva, and E. Soares, "A computational model for developing semantic web-based educational systems," *Knowledge-Based Systems*, vol. 22, no. 4, pp. 302 – 315, 2009.

[3] D. Gasevic, D. Djuric, V. Devedzic, and B. Selic, *Model Driven Architecture and Ontology Development*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[4] P. Tetlow, J. Z. Pan, D. Oberle, E. Wallace, M. Uschold, and E. Kendall, "Ontology driven architectures and potential uses of the semantic web in systems and software engineering", World Wide Web Consortium, Tech. Rep., February 2006.

[5] P. Kogut, S. Cranefield, L. Hart, M. Dutra, K. Baclawski, M. Kokar, and J. Smith, "Uml for ontology development," *Knowl. Eng. Rev.*, vol. 17, no. 1, pp. 61–64, 2002.

[6] K. Baclawski, M. K. Kokar, P. A. Kogut, L. Hart, J. Smith, J. Letkowski, and P. Emery, "Extending the unified modeling language for ontology development," *Software and Systems Modeling*, vol. 1, pp. 142–156, 2002.

[7] K. Falkovych, M. Sabou, and H. Stuckenschmidt, "Uml for the semantic web: Transformation-based approaches," 2003.

[8] S. Cranefield, "Uml and the semantic web," 2001.

[9] E. de Barros Costa, "Um modelo de ambiente interativo de aprendizagem baseado numa arquitetura multi-agentes," Ph.D. dissertation, Universidade Federal da Paraíba, Campina Grande, 1997.

[10] D. L. McGuinness and F. van Harmelen, "Owl web ontology language overview," World Wide Web Consortium, W3C Recommendation, February 2004.

[11] D. Gasevic, D. Djuric, V. Devedzic, and V. Damjanovi, "Converting uml to owl ontologies," in *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*. New York, NY, USA: ACM, pp. 488–489, 2004.

[12] E. de Barros Costa, M. Lopes, and E. Ferneda, "Mathema: A learning environment based on a multi-agent architecture," in *Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, J. Wainer and A. Carvalho, Eds. Springer Berlin / Heidelberg, vol. 991, pp. 141–150,1995.

[13] E. de Barros Costa and A. Perkusich, "Modeling the cooperative interactions in a teaching/learning situation," in *Intelligent Tutoring Systems*, ser. Lecture Notes in Computer Science, C. Frasson, G. Gauthier, and A. Lesgold, Eds. Springer Berlin / Heidelberg, vol. 1086, pp. 168–176, 1996.

[14] M. Fernández, A. Gómez-Pérez, and N. Juristo, "Methontology: from ontological art towards ontological engineering," in *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, 1997.

[15] A. Gmez-Prez, "Knowledge sharing and reuse," in *Handbook of Expert Systems*, J. Liebowitz, Ed. CRC, 1998.

[16] M. F. L´opez, A. G´omez-P´erez, J. P. Sierra, and A. P. Sierra, "Building a chemical ontology using methontology and the ontology design environment," *IEEE Intelligent Systems*, vol. 14, pp. 37–46, January, 1999.

[17] A. Gómez-Pérez, M. Fernández-López, and O. Corcho, *Ontological Engineering*. Berlin: Springer, 2004.

[18] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

[19] H. T. Al-Jumaily, D. Cuadra, and P. M. anez, "Ocl2trigger: Deriving active mechanisms for relational databases using model-driven architecture," *Journal of Systems and Software*, vol. 81, no. 12, pp. 2299 – 2314, 2008

[20] Brown, "An introduction to model driven architecture part i: Mda and today's systems", IBM, Tech. Rep., 2004.

[21] Y. Pan, G. Xie, L. Ma, Y. Yang, Z. Qiu, and J. Lee, "Model driven ontology engineering," in J*ournal on Data Semantics VII*, ser. Lecture Notes in Computer Science. Springer Verlag Berlin, vol. 4244, pp. 57–78, 2006.

[22] A. Kleppe, Warmer, J., and W. Bast, *MDA Explained: The Model Driven Architecture: Practice and Promise*. Indianapolis: Addison-Wesley, 2003.

[23] E. A. Tetlow, "Ontology driven architectures and potential uses of the semantic web in systems and software engineering," W3C, Tech. Rep., 2005.

[24] D. Djuric, D. Gasevic, and V. Devedzic, "Ontology modeling and mda," *Journal of Object Technology*, vol. 4, no. 1, pp. 109–128, Jan. 2005.

Marlos Tacio Silva is PhD student in Computer Science at Federal University of Campina Grande, and member of GrOW (Group of Optimization of Web). His research focus is Multi-agent Systems, Software Engineering and Semantic Web.

Endhe Elias is undergraduate student in Computer Science at Federal University of Alagoas and member of GrOW (Group of Optimization of Web). His research focus is Multi-agent Systems, Software Engineering, Semantic Web and development of Intelligent Tutoring Systems.

Evandro Costa received his PhD in Electrical Engineering from the Federal University of Paraiba (UFPB), Brazil. He is currently an Associate Professor in the Computer Science Institute at the Federal University of Alagoas (UFAL), Brazil and Director of the same institute. He is also a collaborator professor in the graduate program at Federal University of Campina Grande (UFCG), Paraiba, Brazil. His research interests include Multi-agent Systems, Knowledge Representation and Semantic Web, and Artificial Intelligence in Education, where he has published over 100 peer-reviewed papers. He teaches several issues, such as, Artificial Intelligence, Multiagent Systems, Data Mining, and so on.

Ig Ibert Bittencourt is a Professor and Vice-Director at the Computing Institute at the Federal University of Alagoas - UFAL, Brazil, Head of the GrOW (Group of Optimization of the Web) and member of the IEEE, ACM, and Brazilian Computer Society. He teaches issues, such as, Software Engineering, Knowledge Representation and Semantic Web, Artificial Intelligence in Education, and so on. His main research interests are Social Semantic Web and Education, Intelligent Tutoring Systems and Software Engineering in Education.

Heitor Barros is PhD student in Computer Science at Federal University of Campina Grande (UFCG), and member of GrOW (Group of Optimization of Web). In his master thesis, he worked in the design of systems based on Semantic Web Services, focusing on Middleware Grinv, shown in this work. His mains interests are Semantic Web Services, Software Engineering and Knowledge Engineering.

Leandro Dias da Silva has graduation in Computer Science by Universidade Federal de Alagoas (1999), master's at Electrical Engineering by Universidade Federal da Paraíba (2002) , Ph.D. at Electrical Engineering by Universidade Federal de Campina Grande (2006). Currently is at Universidade Federal de Alagoas. He has experience in the area of Computer Science , with emphasis on Petri nets, Component Based Software Engineering, Software Architecture, Product Lines, Reuse, Pervasive Computing and Embedded Systems.

Alan Pedro da Silva bachelor in Computer Science from Federal University of Alagoas (2004), Master in Computational Modeling of Knowledge from the Federal University of Alagoas (2006) and received the Ph.D degree from Federal University of Campina Grande (UFCG) - Brazil, in 2011. Currently he is Professor at the Computing Institute, Federal University of Alagoas (UFAL). Their research interest are Software engineering and artificial intelligence applied to Educational Systems.

Douglas Véras is graduated from the Federal University of Alagoas (UFAL) in Computer Science, he is external member of the GrOW and recently has developed research and development on Digital TV Interactive, specially in t-learning, theme of its conclusion work. Already is a Master's Student Federal University of Pernambuco (UFPE), in line with media research and interaction (TVDi) from course of Computer Science, where also works in partnership with Samsung Digital TV.