Model-driven development of learning objects

Marco Aurélio Graciotto Silva, Ellen Francine Barbosa, José Carlos Maldonado
University of São Paulo, magsilva@icmc.usp.br, francine@icmc.usp.br, jcmaldon@icmc.usp.br

Abstract - **The development of effective learning objects that explores blended learning, collaborative, and open development requires a laborious process. Information created and used at each phase of the process must be manually translated and augmented until achieving a proper product. The issue of systematic development of learning objects can be addressed using model-driven development. In fact, each phase in the development of learning objects requires a specific type of user/developer profile and generates different models. This article defines a model-driven approach for the open and collaborative development of learning objects. The approach herein described uses concept maps, represented as CXL documents, and statechart-based models, which are represented as UML models in XMI documents. Finally, the later model is used to generate learning objects, given a specific platform description. The current implementation supports the generation of slides in LaTeX/Beamer format. The feasibility of the approach is demonstrated using a course on software testing for undergraduate students, with learning objects generated for a context that comprises both traditional classroom and blended learning.**

Index Terms – concept map, learning object, model-driven development, Statecharts, UML profile.

INTRODUCTION

Currently, the development of effective learning objects, which explores blended learning, collaborative and open development, is a laborious process. In general, information created/used at each phase of the process must be manually translated and augmented until achieving a proper product. For example, learning goals should be reused for designing learning objects, educational scenario descriptions should be combined with learning objects, whose deployment in learning management systems should reflect the sequencing established in the learning design. This is a rather simple scenario, but challenging in the pace presently required for education, which requirements are flexible, driven by user's needs and technological devices availability.

The issue of systematic development of learning objects can be addressed using model-driven development. In fact, each phase in the development of learning objects requires a specific type of user/developer profile and generates different models. Instead of relying on a single tool, the hypothesis is that it should be better to use specific tools for each phase and provide transformation rules between each model. The outcomes of each tool, usually a file defined in a domain-specific language, is transformed into a more specific model, until the final set of learning objects is generated. As the entire development is based upon models and their transformations, it is possible to perform the translations more easily, just requiring the modification of specific parameters and the generation of new models.

Motivated by this scenario, this article defines a model-driven approach, called LODM (Learning Object Development Method) for the open and collaborative development of learning objects. It builds upon IMA-CID (Integrated Modeling Approach – Conceptual, Instructional, and Didactic), an approach for developing learning objects using a set of models, each one addressing specific concerns of educational content development [1; 2]. For each model, domain-specific languages are defined using either UML (Unified Modeling Language) profiles or *de facto* standard specification languages. Transformations required to create each model are specified, which are partially supported by LODE (Learning Object Development Environment). The feasibility of LODM/LODE is demonstrated in the development of a course on software testing for undergraduate students. The learning objects have been generated based on LODM/LODE principles, for a context that comprises both traditional and blended learning settings.

This paper is organized as follows. At first, we provide an overview of learning objects development. Next, we summarize model-driven development and briefly describe the potential of its use. Thirdly, we present the IMA-CID approach. The next section defines the model-driven approach for learning objects development – LODM –, and introduces its accompanying tool – LODE. Related work is discussed in the following section. Finally, we summarize contributions and perspectives for further work.

LEARNING OBJECTS DEVELOPMENT

A learning object is an entity, digital or not, that may be used for learning [3]. An entity is often regarded as something static, such as didactic content. However, an effective learning process also requires learning activities (exercises, assessments, discussions) [4]. Learning design plays an important role on learning experiences, guiding the activities and providing some confidence regarding the effectiveness of the learning activity. Considering a technological viewpoint, learning designs should be formally described using Educational Modeling Languages (EMLs) [5], allowing their deployment and execution in Learning Management Systems (LMS). Instances of a learning design should be packaged in a common format for educational context exchange and storage, defining the sequencing, metadata, and associated content, fostering their reuse.

The design of a single unit of learning is an established research topic. Methods based on instructional design [6] were devised for the development of learning objects and units of learning [7; 1]. Personalization techniques are employed to tailor learning objects and management systems to each user profile [8; 9]. Nonetheless, the issue of systematic development of learning objects has not been addressed. Even considering the techniques above, the development of a single learning object still demands a great effort. For instance, if such objects have to be modified for each context, considering the student profile, computational devices, and connectivity, such development could not be feasible.

MODEL-DRIVEN DEVELOPMENT

Learning objects development consists of the progressive development of artifacts, based on tacit or explicit (knowledge) models. Software engineering also follows such an approach, creating models that are iteratively augmented with information until the effective implementation of the model. A common issue is that, once a new model is created, the previous one is abandoned (or, at least, not accordingly updated). Model-driven development is a software engineering technique that tackles this issue, adopting models augmentation and transformation for the entire software life cycle [10]. Thus, modifications are done not in the last model (or even its implementation), but in the model that captures the relevant knowledge regarding the change. For instance, if an issue is discovered about a missing learning objective, the change is done in the first model (analysis model) and propagated to the others by means of (semi-)automatic transformations of subsequent models.

A model is a coherent set of formal elements describing something built for some purpose that is amenable to a particular form of analysis [11]. For instance, learning objects development requires: the analysis of learning objectives goals, activities, content; design; development; implementation (deployment); and evaluation [6]. Each of these concerns can be represented with distinct, yet directly related, models. The relations are drawn upon the augmentation, weaving, and transformation of a simpler or more abstract model to a new one.

Models are represented by domain-specific languages, i.e., a language that adequately represents the information of a given domain. Instead of representing elements using a general purpose language, producing source code (e.g., Java) and class diagrams (e.g., plain UML models), the knowledge is described in a language which the domain experts understand (for instance, concept maps). Actually, the experts in each domain hold responsibility for the artifacts that compose the product (and not the computing technology expert) [12] . Besides, as the expert uses a suitable language to describe the system at hand, the accidental complexity – that one inherent from the restrictions of the language to properly describe a given domain – is reduced, leaving only the essential complexity of the problem [13].

Model transformations are performed by mapping functions that represents the knowledge once retained just by the expert [11]. Such transformations are automatically or semi-automatically executed [12]. For instance, the learning objectives defined by the pedagogical expert must be realized by the content of a learning object. Instead of relying entirely on one person to perform this mapping, a set of rules can be used to bootstrap the design model of the learning object. As there is a formally defined process or tool that realizes the transformation, the risk of losing information from one model (e.g., to not consider a given learning objective) is mitigated.

IMA-CID APPROACH

IMA-CID (Integrated Modeling Approach – Conceptual, Instructional, and Didactic) [1] is a learning design development approach based upon the concept mapping technique [14] and the HMBS (Hypertext Model based on Statecharts) model [15], integrating different modeling aspects related to the development of educational content. The approach aims at the designing of educational modules/units of learning, i.e., concise units of study/learning, composed by theoretical and practical content, which can be delivered to learners by using technological and computational resources [1]. Educational modules can be implemented or realized as learning objects of varying granularity: the whole module can be defined as a learning object, or parts of it can be realized by a (reusable) learning object or exported as learning object itself.

Three perspectives are considered while designing a unit of learning: conceptual, instructional, and didactic [16]. For each of them, a model is defined, representing, respectively, the main concepts of the domain and their relationships, definition and organization of additional information regarding those concepts, and sequencing for the presentation of the concepts and related information. In the following subsections, all the models are described.

I. Conceptual Model

IMA-CID is based upon learning objectives defined in a conceptual model for the domain of the learning object. The model is represented by a concept map [14]. Besides the intrinsic elements represented in the map (concepts and specific domain relationships between them), two types of relationship should be identified: taxonomy (type-of) and composition (part-of).

II. Instructional Model

The instructional model defines information items and instructional elements regarding the concepts identified in the conceptual model. The main purpose of information items is to describe the concepts (identified in the conceptual model). IMA-CID uses the Component Display Theory (CDT) [17] to represent them as concepts, facts, procedures, and principles. Besides information items, the instructional model may contains instructional elements, which are classified into explanatory (complementary information that further explains a given concept), exploratory (foster the navigation within the unit of learning/between learning objects) and evaluative (assess the learner's knowledge and progress regarding the learning objectives) [1].

Information items and instructional elements are organized in a HMBS model, which uses the structure and execution semantics of Statecharts to specify the structural organization and browsing semantics of hypertexts [15]. Structural and composition relationships, defined in the conceptual model, are realized using nested-states while domain specific relations are usually represented by specific transitions between states or regions within each state. Information items and instructional elements are represented within states and its regions.

III. Didactic Model

The Didactic Model is responsible for the establishment of prerequisites and sequences of presentation among conceptual and instructional elements. It is modeling using an extended version of HMBS, which defines the notion of DD (Dynamically Defined) states [1]. A DD-state defines a transition between every sibling-state that can be activated by any event. In practice, it allows the open navigation between sibling states/concepts, providing support for the definition of dynamic contexts of learning [1].

LODM AND LODE

LODM (Learning Object Development Method) is an extension to IMA-CID. Its main feature is the adoption of a model-driven development approach for learning objects, which is implemented using models (based on the aforementioned IMA-CID approach) and transformations. Considering this perspective and our experience on the use of the technique, modifications to each model were defined to ease the use of models, adopting simpler, yet complete, models, and improved tooling support. In this regard, we used commercially-supported languages (such as concept maps and UML models) and, for the transformations, we developed a tool: LODE (Learning Object Development Environment).

I. Conceptual Model

Concept maps can be used to realize the knowledge regarding a given topic [18]. Although it can be obtained from tacit knowledge (the professor's mind), a common strategy is to extract the main concepts and their relationships from existing didactic content. The reuse of previously developed material is recommended (after all, one goal of learning objects is the reuse). Indeed, most of the learning objects created so far using IMA-CID are based on textbooks and slides used in classes. Thus, two techniques can be adopted: annotation of the content or data mining.

Annotation is the technique adopted for all learning objects developed using IMA-CID until now. It is straightforward: relevant concepts are identified (e.g., nouns used in titles) and instances of such concepts in the text are candidate relations. However, this activity is error-prone (due the length of the source text) and expensive (given the time required). An alternative is to employ data mining techniques, automatically identifying the main concepts and relationships [19]. Data mining is the approach adopted by LODE. Given a didactic material as source, its text is extracted, preprocessed, and analyzed: words that are frequently used are candidates for a concept. Sentences that use such words can also be identified, providing additional aid to the author regarding relationship definitions. All these functionalities are fully implemented by LODE. For instance, considering a previous course on Software Testing [20], the concepts more frequently used are those shown in Table 1. Although it still demands some later action, either to review or to include concepts not automatically found, it relieves the professor of identifying the trivial ones. At the same time, it avoids errors (such as the omission of concepts or relations that are trivial to the professor, but relevant for the learning object and the students).

Words (1-gram)	**Words (2-gram)**
testing (314), test (244), criterion (199), mutation (136), suite (125), case (102), mutant (92), node (63), use (63), program (59), use (63), program (59), product (56) path (53), fault (52), variable (52), ….	test suite (122), test case (87), testing criterion (70), mutation testing (59), mutation operator (29), structural testing (28), testing requirement (25), data flow (24), equivalence partition (23), mutation score (23), equivalent mutants (22), control data (21), testing technique (18), testing tool (16), …

TABLE 1. TERMS EXTRACTED FROM A DIDACTIC MATERIAL AND THEIR FREQUENCY.

After the extraction of the concepts and relationships, it is possible to build the concept map, as show in Figure 1. For now, this is manually done, using the software CmapTools [21]. It supports the collaborative editing of concept maps and can export a concept map in an XML document (CXL – Concept Mapping Extensible Language) [22], which is suitable for the purpose of automatic transformation of the model-driven development.

II. Instructional Model

Currently, IMA-CID adopts a formal model HBMS. However, after the development of several units of learning using the technique, we identified that some features regarding the model are not used. Actually, comparing it with the base technique, Statecharts, only the DD-state is required. Thus, LODM focuses on using a model that closely resembles Statecharts, defining UML profiles to educational-specific information (concepts, information items, and instructional elements). Not only this would suffice to build the instructional model, but it would also allow the usage of tools that supports UML and common data models and interchange formats, such as XMI (XML Metadata Interchange).

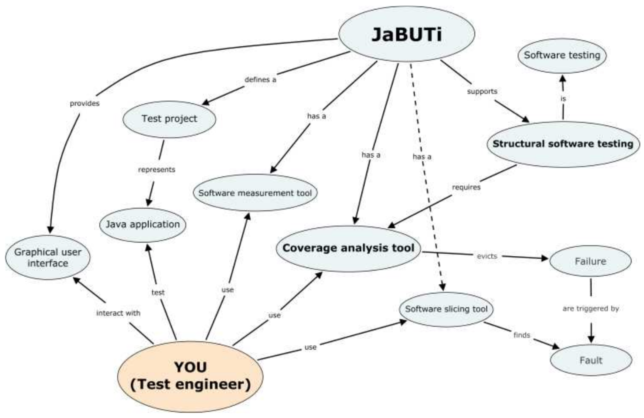


FIGURE 1. CONCEPT MAP ABOUT A SOFTWARE TESTING TOOL (JABUTI).

LODM also leverages IMA-CID support to information items and instructional elements description. It defines a UML profile, named IMA-CID profile, to allow the extension of information item types, relieving the restriction to CDT-based elements. Moreover, the profile defines a different behavior for instructional elements: any model element that is not an information item. The classification previously established by an instructional element (explanatory, exploratory, and evaluative) is now represented by a stereotype, and can be used for any IMA-CID model element.

UML is a general-purpose modeling language that provides abstractions that support a wide variety of concepts in known problems domains, and language extensions mechanisms that can be used to define domain specific-languages [13]. Such extensions are defined using UML profiles: set of stereotypes, restrictions, and properties (tagged values) that allow the specialization of the UML metamodel for a specific purpose [23]. In UML terms, a profile is represented as the specialization of a package, which is associated to stereotypes. For each stereotype, restrictions and tagged values can be specified.

The IMA-CID profile (Figure 2) comprises of two stereotypes: `information item` and `instructional element`. Both inherit from `IMA-CID element`, an abstract stereotype that labels the element as one that belongs to an IMA-CID model.

As previously mentioned, a stereotype can be associated with tagged values. Two tagged values are defined for an IMA-CID element: `name` (identifier of the element) and its `type` (as defined by an ontology or, at least, the media type − MIME type). The current IMA-CID's semantics regarding an information item is represented by the (concrete) stereotype `Component display theory element`. However, it is not restricted to only this stereotype. In fact, other information models can be considered while building the instructional model. For instance, it is possible to use a concept map as part of the model (stereotype `Concept mapping`).

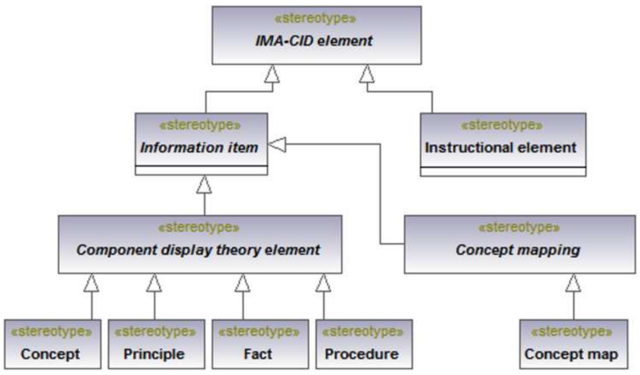


FIGURE 2. IMA-CID PROFILE – IMA-CID ELEMENT STEREOTYPE.

Any model element defined within a region of a state and that cannot be assigned to an `Information item` stereotype should be assigned an `Instructional element` stereotype. While most of the elements can be represented by the framework currently provided (as a CDT- or concept map-element), some cannot (e.g., an exercise).

Differently of the current IMA-CID models, the classification of explanatory, exploratory or evaluative are not exclusive to instructional elements, but available to any IMA-CID element. A new stereotype, `Educational Goal`, can be assigned to any `IMA-CID element`, denoting the purpose of the model element. Therefore, any element (and not just instructional elements) can have its goal defined. The current implementation of the profile (Figure 3) defines three concrete instances of the abstract `Educational goal` stereotype: `Explanatory`, `Exploratory`, and `Evaluation`. Afterwards, each concrete information item stereotype can restrict the values assignable to the tagged values (e.g., a fact cannot be used for assessment while a concept map can be used for any goal).

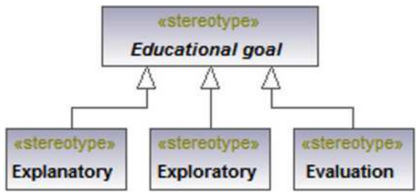


FIGURE 3. IMA-CID PROFILE - EDUCATIONAL GOAL STEREOTYPE.

Using the aforementioned profile, a unit of learning could be described as shown in Figure 4. It defines the instructional model for a module of a Software Testing course about JaBUTi (a software testing tool). Six states were specified: one for the unit itself (JaBUTi) and five sibling states: Overview, General Information, Architecture, Licensing, and Resources. Within each state, at least one region is defined (hashed lines are used to divide the state into more than one region). Each region represents an information item or an instructional element.

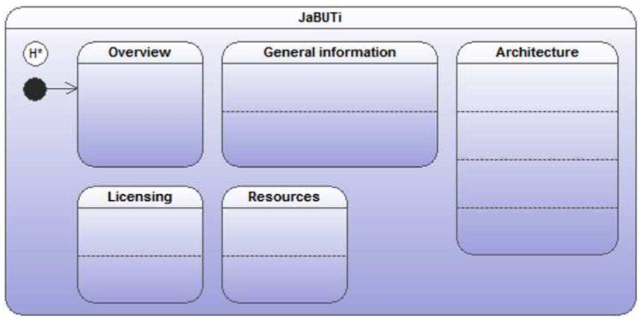


FIGURE 4. INSTRUCTIONAL MODEL FOR A SOFTWARE TESTING COURSE.

The rendering of a stereotype for a given UML element will depend upon the software used to model (the standard itself does not rule against this aspect). The tool we are currently using, Altova UModel [24], does not allow the definition of a new shape or any other identifier for stereotypes applied to a state's region.

Nonetheless, the stereotype and related tagged values can be defined and are preserved in the model, which is sufficient for the model-driven engineering goal. For instance, in Figure 5, the state selected, `General Information` has two regions. The first one (which is selected), is an information item (`Concept` stereotype) that has the didactic goal of explaining (as defined the by `Explanatory` stereotype). Two tagged values, as required by `Information item` stereotype, are defined: `type` and `id`. The type is the MIME type of the element (plain text). The identifier corresponds to the URL of the page that contains the data.

The use of the aforementioned profile proceeds as follows. The concept map, more specifically the CXL documents created from the model, is used to define the states. For every concept, a state is defined. If the concept is related to another concept by a hierarchy (`type-of`) relation, the state is created as child of the first. If the concept is related to another concept by a composition (`part-of`) relation, the state is created as a sibling. For concepts associated by specific-relations, information items should be identified. The elements identified must be later specified. We adopted a wiki for this purpose, fostering the collaborative development of the information items, under the assumption that, in a constructionist learning process, learners will use the wiki to share their findings [25], providing new information items and instructional elements (facts, procedures, examples, and so on).

III. Didactic Model

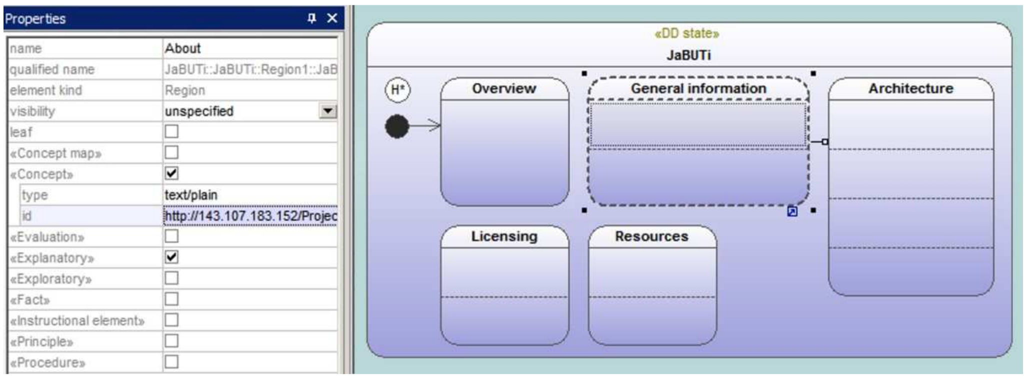
In order to represent the extended HMBS, we created a second profile, SHMBS, which modifies the behavior of the state machine. It defines the DD stereotype, which implements the *DD* (*Dynamically Defined*) *state*. This stereotype can be defined for UML elements of the type `BehaviorStateMachines::State`. The model in Figure 5 defines the state `JaBUTi` as a *DD state*, as denoted by the stereotype `DD state` defined on the top of state.

IV. Generation

After defining the didactic model, it can be exported as an XMI document [26]. This document will be translated into the final presentation (slides). We defined a set of transformation rules to generate LaTeX source files, using the Beamer package. Briefly, each state corresponds to a set of slides. From a given slide, the user can navigate: (1) to the first slide of any sibling state; (2) to the previous or next slide of the current state; (3) to the first slide of the parent state; (4) to the first slide of any child state; and (5) to the first slide of any state that is reachable from the current state (not necessarily a sibling state). The generated files are then compiled, creating a set of PDF files, which can be used as learning objects and deployed in a learning management system.

RELATED WORK

Souza et al. [27] define a domain-specific language that allows the customization of learning objects. The focus of their research is personalization, creating learning objects whose content and activities depends upon the learner profile (level of knowledge and cultural elements).


Properties

name	About
qualified name	JaBUTi::JaBUTi::Region1::JaB
element kind	Region
visibility	unspecified
leaf	☐
«Concept map»	☐
«Concept»	✓
type	text/plain
id	http://143.107.183.152/Projec
«Evaluation»	☐
«Explanatory»	✓
«Exploratory»	☐
«Fact»	☐
«Instructional element»	☐
«Principle»	☐
«Procedure»	☐

FIGURE 5. MODEL FOR PART OF THE SOFTWARE TESTING COURSE AND DEFINITION OF STEREOTYPES FOR A STATE.

MD2 [28] defines a model-driven approach for didactic material creation. It uses a different, but however complementary perspective: while LODM supports the creation of learning objects from the ground-up, MD2 supports the creation of didactic material considering a set of (meta-)requirements such as type of material, learning objectives, time restriction, supporting medium, etc. Actually, MD2 creates composition of learning objects: the models are requirements for the composition. For instance, such models could be used to select part of the instructional model considering the educational context.

CONCLUDING REMARKS AND FURTHER WORK

In this paper, we briefly describe a model-driven approach for the open and collaborative development of learning objects. LODM/LODE supports the development of learning objects from learning objectives definition until content generation. It empowers users, as they use specific-domain languages and tools that are commercially supported (thus lowering the learning curve). Although IMA-CID provides sufficient mechanisms for the development of simple learning objects, the lack of support for different types of information items (such as concept maps) and scalability issues imposes a burden to instructional designers and professors. The shift to model-driven engineering and evolution of the inherent information model (represented by information items and instructional elements), as defined by LODM and its associated environment, LODE, allow the development of bigger and more complex learning objects. The course on software testing, developed according to the ideas discussed in this paper has: 335 concepts and 392 associations for the conceptual model; 141 states, 521 information items, and 73 instructional elements for instructional and didactic models; and 576 slides compiled from a set of 241 files generated from the didactic model.

Currently, the transformations regarding the conceptual model are implemented by LODE, and the remaining ones are fully described by LODM. Further work will be directed to implement the remaining transformations and the evaluation of generated learning objects. Moreover, considering that the educational process depends upon its context, transformations that consider the user's profile, as defined by ontologies, will be explored in further works.

ACKNOWLEDGMENT

We would like to thank the Brazilian funding agencies – CNPq, CAPES and FAPESP – for their support.

REFERENCES

[1] Barbosa, E. F. and Maldonado, J. C. 2006. *An Integrated Content Modeling Approach for Educational Modules.* In: IFIP World Computer Congress.Santiago, Chile: Springer, Vol. 210, pp. 17-26.

[2] Barbosa, E. F. and Maldonado, J. C. 2011. *Collaborative Development of Educational Modules: a Need for Lifelong Learning.* In G. D. Magoulas. *E-Infrastructures and Technologies for Lifelong Learning: Next Generation Environments.* London: IGI Global, pp. 175-211.

[3] IEEE LTSC. *IEEE 1484.12.1 - Learning Object Metadata*. 2002.

[4] Martinez-Ortiz, I., Sierra, J. and Fernandez-Manjon, B. 2009. *Authoring and Reengineering of IMS Learning Design Units of Learning,* IEEE Transactions on Learning Technologies, Vol. 2, pp. 189-202.

[5] Rawlings, A. et al. 2002. *Survey of Educational Modelling Languages.* In: CEN/ISSS Workshop on Learning Technologies.

[6] Dick, W., Carey, L. and Carey, J. O. 2006. *The Systematic Design of Instruction.* USA: Pearson, p. 376.

[7] Baruque, L. B. and Melo, R. N. 2004. *Learning Theory and Instruction Design Using Learning Objects.* Journal of Educational Multimedia and Hypermedia, Vol. 13, pp. 343-370.

[8] Heiyanthuduwage, S. R. and Karunaratne, D. D. 2007. *A Learner Oriented Ontology to Make Effective Learning Management Systems.* In: International Conference on Digital Information Management, Lyon, France, pp. 476-481.

[9] Gaeta, M., Orciuoli, F. and Ritrovato, P. 2009. *Advanced ontology management system for personalised e-Learning.* Knowledge-Based Systems, Vol. 22, pp. 292-301.

[10] Stahl, T. and Völter, M. *Model-Driven Software Development.* Great Britain: Wiley, 2006. p. 428.

[11] Mellor, S. J., Clark, A. N. and Futagami, T. 2003. *Model-Driven Development.* IEEE Software, Vol. 20, pp. 14-18..

[12] Selic, B. 2003. *The pragmatics of model-driven development.* IEEE Software, Vol. 20, pp. 19-25.

[13] France, R. and Rumpe, B. 2007. *Model-driven Development of Complex Software: A Research Roadmap.* In: Future of Software Engineering. Minneapolis, USA: IEEE,. pp. 37-54.

[14] Novak, J. D. 1990. *Concept mapping: A useful tool for science education.* Journal of Research in Science Teaching, Vol. 27, pp. 937-949.

[15] Oliveira, M. C. F., Turine, M. A. T. and Masiero, P. C. 2001. *A statechart-based model for hypermedia applications.* ACM Transactions on Information Systems (TOIS), Vol. 19, pp. 28-52.

[16] Barbosa, E. F., Maldonado, J. C. and Ricarte, L. M. 2002. *Learning materials: Towards the establishment of guidelines for domain modeling.* In: Conference on Informatics Curricula, Teaching Methods and Best Practice, Florianópolis, Brazil, pp. 192-206.

[17] Merrill, M. D. 1983. *Component Display Theory.* Lawrence Erlbaum.

[18] Coffey, J. W., et al. 2002. *.A Concept Map-Based Knowledge Modeling Approach to Expert Knowledge Sharing.* In: IASTED International Conference on Information and Knowledge Sharing,Virgin Islands, USA.

[19] Pérez-Marín, D., et al. 2007. *Automatic Generation of Students' Conceptual Models from Answers in Plain Text.* [ed.] Cristina Conati, Kathleen McCoy and Georgios Paliouras. s.l. : Springer Berlin / Heidelberg, 2007. Vol. 4511, pp. 329-333.

[20] Vincenzi, A., et al. 2010. *Functional, Control and Data Flow, and Mutation Testing: Theory and Practice*. In: Borba, P. et al., *Testing Techniques in Software Engineering.* Springer, Vol. 6153, pp. 18-58.

[21] Cañas, A. J. et al. Navarra. 2004. *CmapTools: A Knowledge Modeling and Sharing Environment.* In: International Conference on Concept Mapping. Spain.. Vol. 1, pp. 125-133.

[22] Cañas, A. J. et al. 2006. *KEA: A Knowledge Exchange Architecture Based on Web Services, Concept Maps and CmapTools.* In: International Conference on Concept Mapping. San José, Costa Rica, Vol. 1, pp. 304-310.

[23] Object Management Group, 2010. *UML - Infrastructure*. Version 2.3.

[24] Altova. UModel. 2005.

[25] Forte, A. and Bruckman, A. 2007. *Constructing text: Wiki as a toolkit for (collaborative?) learning.* In: International Symposium on Wikis. Montreal, Canada: ACM, pp. 31-42.

[26] Object Management Group. 2007. *MOF 2.0/XMI Mapping.*

[27] Souza, M. F. C., Castro-Filho, J. A. and Andrade, R. M. C. 2010. *Model-Driven Development in the Production of Customizable Learning Objects.* In: International Conference on Advanced Learning Technologies, Sousse, Tunisia, pp. 701-702.

[28] Padrón, Carmen L., Diaz, P. and Aedo, I. 2006. *MD2 Method: The Didactic Materials Creation from a Model Based Perspective.* In European Conference on Technology Enhanced Learning (EC-TEL). Crete, Greece, pp. 366-382.