A domain-specific language for Virtual Classrooms

Article *in* International Journal of Metadata Semantics and Ontologies · October 2014

DOI: 10.1504/IJMSO.2014.065444

4 authors:

David Gañán Jiménez
Universitat Oberta de Catalunya
39 PUBLICATIONS 218 CITATIONS

Ana-Elena Guerrero
Universitat Oberta de Catalunya
40 PUBLICATIONS 172 CITATIONS

Josep Prieto-Blazquez
Universitat Oberta de Catalunya
37 PUBLICATIONS 392 CITATIONS

Jordi Conesa
Universitat Oberta de Catalunya
110 PUBLICATIONS 494 CITATIONS

A domain-specific language for Virtual Classrooms

David Gañán*, Ana-Elena Guerrero-Roldán, Josep Prieto-Blázquez and Jordi Conesa

Estudis d'Informàtica i Multimedia,
Universitat Oberta de Catalunya,
Rambla del Poblenou, 156,
08018 Barcelona, Spain
Email: dganan@uoc.edu
Email: aguerreror@uoc.edu
Email: jprieto@uoc.edu
Email: jconesac@uoc.edu
*Corresponding author

Abstract: This paper proposes the creation of a model-driven development approach on top of a Virtual Learning Environment (VLE) in order to automate the process of creating Virtual Classrooms. In particular, a Domain-Specific Language (DSL) has been defined for the creation of Virtual Classrooms reusing a specific ontology for Virtual Classrooms. The DSL has been validated in a real case study: the creation of Virtual Laboratories at the Open University of Catalonia. The validation has shown that the approach enhances the reusability of Virtual Classrooms for the upcoming courses and the DSL allows describing Virtual Classrooms graphically at a high abstraction level. In addition, classrooms can be easily modified and automatically exported to other learning systems like Moodle.

Keywords: domain ontologies; Virtual Classrooms; domain-specific languages; model-driven development.

Reference to this paper should be made as follows: Gañán, D., Guerrero-Roldán, A-E., Prieto-Blázquez, J. and Conesa, J. (2014) 'A domain-specific language for Virtual Classrooms', *Int. J. Metadata, Semantics and Ontologies*, Vol. 9, No. 4, pp.312–323.

Biographical notes: David Gañán is a Computer Science Engineer and Master in Multimedia Applications. He works at Universitat Oberta de Catalunya (UOC) since 1999 taking different roles in Computer Science subjects. He participated in the design and creation of the Microsoft .NET master area, developed many academic projects inside the e-learning field, and participated in the Alice European project for the construction of an adaptative e-learning system. Apart from UOC activities, he works as a Freelance Independent Professional for third parties. He has collaborated with several important companies in the software development area like Microsoft, Bitmakers, Raona, Itequia and WorldSensing.

Ana-Elena Guerrero-Roldán is a Bachelor in Pedagogy from Ramon Llull University, specialist in Online Education and Master in Information and Knowledge Society. In 2011, she received her PhD focused on Adaptive Learning Technology from the Open University of Catalonia. Since 2003, she's been working as a Lecturer in the same university, in the Computer Science Bachelor degree. Her research is centred on the Technology Enhance Learning area, which is based on Virtual Learning Environments (VLEs) for higher education that respond to the users' needs both from personal and teaching perspectives. She has participated in several national and international research projects.

Josep Prieto-Blázquez completed his PhD in Computer Science in 2009 from the Universitat Oberta de Catalunya and received his Master's degree in Computer Science from the Universitat Politecnica de Catalunya in 1993. Since 1998, he has worked as a Lecturer in Computer Science, Multimedia and Telecommunication Department of at the Universitat Oberta de Catalunya and he has been a Dean, in the same department, since 2013.

Jordi Conesa is a PhD in Software Engineering for the Technical University of Catalonia and works as Assistant Professor of Information Systems at Universitat Oberta de Catalunya. His research interest concerns the areas of conceptual modelling, ontologies, semantic web, knowledge-based systems and e-learning. His long-term goal is to develop methodologies and tools to use ontologies effectively in several application domains, such as conceptual modelling, software engineering and e-learning. He has authored over 40 research papers, has participated in nine research projects and has contributed in the organisation of some international conferences, such as the ER 2008 or the INCOS.

1 Introduction

Several standards and specifications are being updated to improve the teaching and learning process and they can be applied to different e-learning systems, but yet not all of them are as useful as teachers expected. Standards and specifications need to be improved if they are to be implemented in a real VLE. When e-learning organisations need to build a VLE, it becomes evident that some of these standards are not really interoperable, reusable and fully supported by any VLE. E-learning systems are evolving quickly, but the real benefits are not keeping pace. The main problems are related to the import and export processes of the previous classrooms and learning resources; the distribution and reusability of the learning processes: resources, spaces and roles.

Throughout the last decade, a set of model-driven technologies have appeared in the field of software engineering that follow a model-driven architecture (OMG, 2013). These technologies support the creation of applications at a higher level of abstraction and use models instead of source code. There are many benefits of using a model-driven philosophy, such as the following: it simplifies the interoperability between different tools/specifications/ standards and facilitates the design and implementation of information systems. Using a Model-Driven Development (MDD) approach in big domains may be very difficult and costly since creating a model that takes into account all the necessary information may be harder work than implementing the system from scratch. However, using MDD in small domains is easier and it has been tested and is useful (MDA, 2003).

In the context of e-learning, the creation of Virtual Classrooms tends to be a non-intuitive and manual process, which, in most cases, is not performed by teachers. The fact that teachers are not directly involved in the creation of their classrooms might produce errors and pedagogical mismatches. It is necessary to create mechanisms that allow Virtual Classrooms to be implemented from a higher abstraction level. Additionally, these mechanisms may take into account information from past semesters or, in some cases, from other institutions. Since Virtual Classrooms define a small context within a VLE, the problem can be easily attacked from an MDD perspective. In addition, the structure of the domain fits very well with MDD. The possibility of using a graphical notation to define the classroom specification will allow teachers with low technical knowledge to define their own classrooms.

Therefore, the main objective of this paper is to create a framework to specify and implement Virtual Classrooms that use an MDD philosophy and reuse previous domain ontologies. We propose to use a domain ontology as a domain model of our MDD approach due to the benefits it provides: a consensual and unambiguous definition of concepts and relationships of the domain; an easy way of reusing and sharing Virtual Classrooms; and the ability to make the Virtual Classrooms understandable for a computer program. Following this approach, an MDD system created on top of a VLE may enhance the interoperability and reusability of all the elements involved in e-learning, and it may also facilitate the modification and the creation of Virtual Classrooms in other systems.

The feasibility and utility of the system is validated by means of a real case study. In this case, study has been implemented and tested on a system to automatically create Virtual Laboratories from their specifications (reusing on an existing ontology) in the context of the Open University of Catalonia (UOC; http://www.uoc.edu).

The rest of the paper is structured as follows: Section 2 reviews the current related work; Section 3 describes the MDD framework elements which are used to model Virtual Classrooms and to translate these models into a real implementation of the Virtual Classroom; Section 4 explains the application of this framework into a real case; the conclusions and further work of the research presented are summarised in Section 5.

2 State of the art

We are not the first to propose using MDD frameworks in e-learning. In this section, some of the relevant related works are reviewed that use an MDD philosophy in order to facilitate the design of learning processes or activities and to automate learning processes. Later, the use of ontologies is discussed in e-learning to contextualise the domain model required for our approach and how it is used.

2.1 MDD frameworks

In e-learning, the MDD approach has been used mainly to automate learning processes and to support the instructional design. According to the automation of learning processes, Rius et al. (2010), Rius et al. (2013) and Rius et al. (2014) present an ontology-driven system that supports the specification of learning processes and automates their implementation from the processes' specifications. A system such as this allows defining the learning processes in three different levels of abstraction using a graphical language similar to BPMN. The first level corresponds to the patterns of learning processes, which define the semantic of the learning processes in a way that is independent of any institution and VLE. The second level defines the learning processes by adapting the patterns of the previous level to the educational institution where they will be used. Finally, the most concrete level of abstraction is used to define the learning services, which are the adaptation of the learning processes to a given VLE.

According to the use of MDD approaches in the support of instructional design, the leading works are Dodero et al. (2007) and Dodero et al. (2010). They present a generative, model-driven engineering approach that allows the creation and adaptation of competence development programs from families of available learning components. The proposed system includes a general architecture based on semantic web services and two ontologies. One is based on LOM (IEEE, 2002) for describing learning resources and the other is based on human resources competencies for describing the existing and targeted competencies of learners and resources. Later, in Dodero et al. (2012), an MDD framework has been created to design learning design courses and to export them into languages and formats of common learning environments and course players.

All these works are similar to ours in the sense that they use an MDD approach and define knowledge at different levels of abstraction in order to separate the conceptual information from the implementation details. Such separation facilitates interoperability, reusability and automation. Apart from that, the difference between our proposal and theirs is mainly the pursued objective; whereas these works are focused on supporting learning process and activities, our work is focused on supporting learning environments (Virtual Classrooms). In fact, our work may complement the presented ones because an ideal MDD approach would take the three aspects of learning environment, learning processes and learning courses into account.

Abdallah et al. (2008) is another related work that focuses on supporting learning environments. In this work, the authors try to create Virtual Classrooms from their specification at a conceptual level. Unfortunately, this work is in its preliminary stages and it has not been fully implemented. In particular, it lacks the adaptation to different VLE and it does not provide graphical facilities in order to create the specifications of Virtual Classrooms. This last functionality is paramount in real applications since teachers with no technical knowledge should be able to create and fine-tune their classrooms.

2.2 Use of ontologies

There has been a great effort made in the semantic web community to provide specifications, standards and ontologies to facilitate semantic processes on the web (Wilson, 2004). In the particular field of e-learning, quite a few ontologies (Wilson, 2004; Babic et al., 2008) and related standards (Berlanga and Garcia, 2005) have been defined so far.

Some works use ontologies to specify the metadata of learning resources in a way that improves the LOM standard in the semantic provided by the metadata and by reducing the ambiguity present in the standard (Rodríguez et al., 2009; Sánchez-Alonso et al., 2007; Ghebghoub et al., 2008; Ghebghoub et al., 2010). As an example, in Pahl and Melia (2006) it is introduced as an ontology-based semantic modelling framework that addresses subject domain modelling, instruction modelling and interoperability aspects in the development of complex reusable learning objects. They remark on the benefits of semantic modelling for learning object assemblies within the context of standards such as SCORM (Dodds and Thropp, 2006) and LOM. In Jovanovic et al. (2006), it is also created as a framework for a learning object context ontology that leverages a range of other kinds of learning ontologies (e.g. user modelling ontology and domain ontology). Regarding the use of ontologies to improve semantic richness of IMS-LD specification (IMS Global Learning Consortium, Inc., n.d.), there is the work by Amorin et al. (2006), who had created an ontology for describing the semantics for level B of the Learning Design specification, supporting intelligent agents to automate the tasks involved during run-time. But none of them have proposed a general model to create and reuse Virtual Classrooms following an MDD approach that takes into account all the learning processes involved (or the whole learning process).

3 The MDD framework for dealing with Virtual Classrooms

In order to use an MDD approach for modelling classrooms and generating their implementation automatically from their models, we have created a framework with the following components:

- A *domain ontology* for representing most of the facets of the Virtual Classroom domain. The ontology will include, among others, general information about the Virtual Classroom, the competencies to be acquired, the resources it contains, the activities that will be done in the teaching related to the classroom and the different persons that will have access to the classroom and with what role. Implementation information is not included in the ontology. A particular classroom may be represented as an instance of such ontology.
- A *graphical language* for allowing a graphical representation of all the information related to Virtual Classrooms. Each element of the graphical notation will be associated with elements of the domain ontology. The language represents graphically the Virtual Classrooms defined in the domain ontology. Such graphical representation aids teachers in the creation of new Virtual Classroom specifications and in the visualisation of existing representations of Virtual Classrooms.
- A *set of transformation rules* for generating the implementation of the classroom in a VLE directly from its specification.

The three components have been integrated by implementing a system that uses (1) the graphical language as interface with users; (2) the domain ontology to store all the relevant information about Virtual Classrooms, to guarantee domain constraints satisfaction and to aid users in the creation of classrooms; and (3) the transformation rules that are used in order to implement a Virtual Classroom from its models. The three components and the integrating system are explained in more detail in the next subsections.

3.1 An ontology for specifying Virtual Classrooms

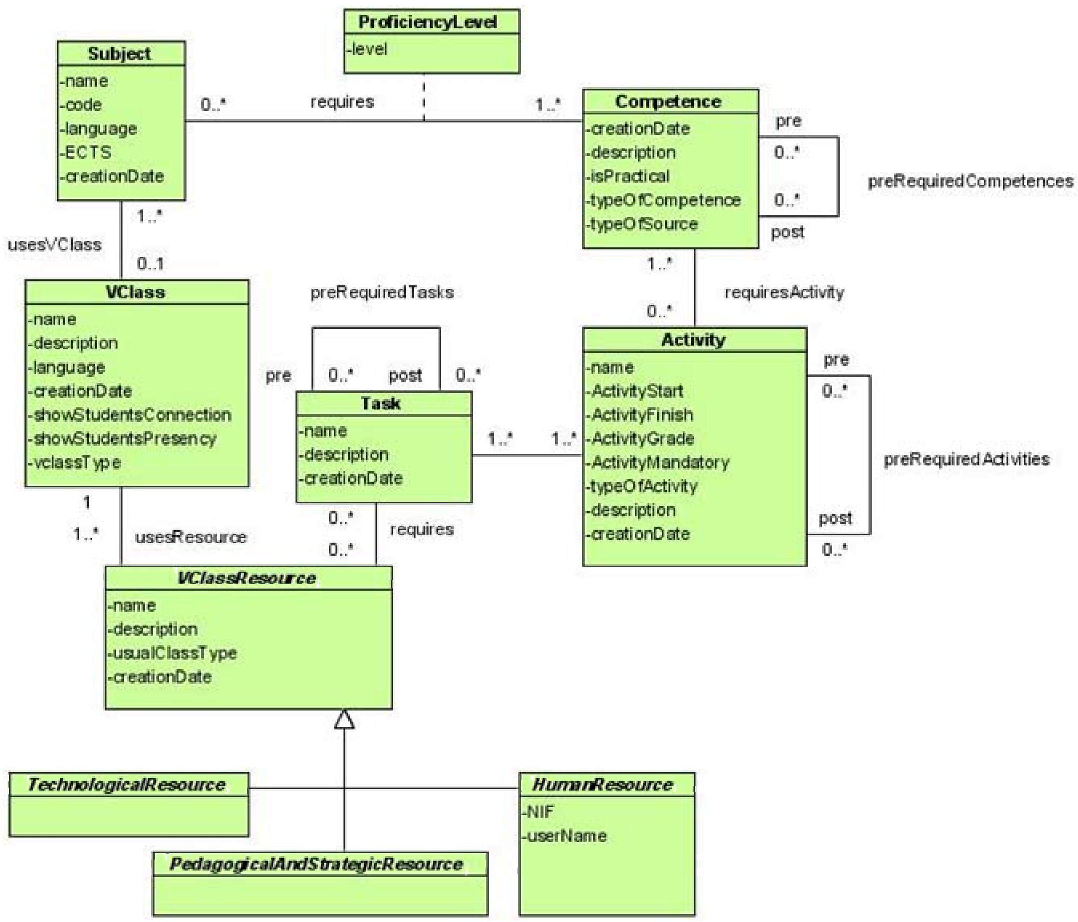
One of the main pieces in MDD approaches is a domain model, usually called a meta-model. It is used to represent the models that define the system to be created. One possible option was to create an ad hoc domain model for the system being developed, but we preferred to use an existing domain ontology as a meta-model. Readers may wonder why using a domain ontology is a domain model. We believe that ontologies fit especially well to MDD as domain models because their characteristics may improve several facets of the MDD system. Some of these improvements may be the following:

- It may facilitate the reusability and the sharing of knowledge because the basic knowledge of the ontology has been agreed by several domain experts.
- It provides guarantees about the quality and validity of the domain model because the ontology has been created and accepted by different members of the domain community instead of by a small group of people.
- It may facilitate the use of the system by users because the domain concepts and relations defined in domain ontologies are named after the terms used in the domain and may also contain synonymous information that may facilitate the users' comprehension.
- It facilitates satisfying the consistency of the domain model since inference engines can be used to check integrity constraints of the domain model.
- It simplifies the transformation rules since the knowledge of the ontology is defined without ambiguity.

The chosen ontology is called *VClass* and allows dealing with Virtual Classrooms. The most relevant fragment of the ontology is depicted in Figure 1 and is described below.

The main element of the ontology is the Virtual Classroom itself, which is known as *VClass*. It is composed of a set of different resources known as *VClassResources*. The ontology describes a wide hierarchy of *VClass Resources*, which can be grouped in three types: human resources, pedagogical and strategic resources and technological resources. These types of resources are represented in the model as three abstract subclasses of *VClassResource* (see Figure 1). The specific virtual class resources will be the children of the corresponding one, depending on its type. For example, student, staff and teacher would be instances of *HumanResource*, while virtual machine, forum, blog and software would be instances of *TechnologicalResource* and evaluation, learning methodology and virtual library would be instances of *PedagogicalAndStrategicResource*. As each specific resource would normally have different and specific properties, these abstract classes do not define properties in the diagram (except from human resources as the identification ID and username common for each subchild).

Figure 1 Conceptual model for the main entities from the VClass ontology (see online version for colours)



The subject is also another important concept within the ontology, because it is the context where the virtual class is used and defines a list of competences to acquire it. Each competence, especially those which are practical, has a list of activities that must be realised in order to achieve it. Activities are composed of a list of tasks to be accomplished, which in turn define a set of resources needed to fulfil that task. By navigating such relationships from *Subject* to *VClassResource*, we can calculate the set of resources needed for a specific subject.

Subjects might have an associated *VClassi*, while the same *VClass* can be reused in different subjects. Note that the *VClass* related to a subject should contain all or part of the *VClassResources* needed to complete the tasks involved in the activities, which are related to the subject's competences (García-Barriocanal et al., 2012). This is not guaranteed by the model itself, so it is a condition that the editor tool must ensure.

3.2 A graphical language for modelling Virtual Classrooms

While classes and relationships of the VClass ontology provide general information about the domain, instances specify concrete Virtual Classrooms within a given context, such as the Virtual Classroom of the subject titled 'Introduction to Databases' in the first semester of 2013. Therefore, instantiating the ontology is a necessary step to specify concrete Virtual Classrooms. However, the direct instantiation of the ontology is rarely usable and prone to error, making it very difficult to add new data to the ontology and to edit the already instantiated data. In order to simplify the process of instantiating the ontology and of editing its instances, we propose using a Domain-Specific Language (DSL). This language should be able to represent the information conceptualised, without ambiguities, in the ontology.

Table 1 Some graphical elements from the VClass notation (see online version for colours)

	Subject		Human resource	Coordinator teacher		Technological resources	Automatic assessment tool	
	Competence			Theory teacher			Blog	
	Practical activity			Lab teacher			Chat	
	Task			Student			Forum	
VClass	Databases VClass		Pedagogical and strategic resources	Learning methodology			Remote laboratory	
	Math's VClass			Manual			Simulator	
	Network VClass			Evaluation			Software	
	Computer architecture VClass			FAQ			Students list	
VClass	Operating systems VClass			Software support			Virtual library	
	Programming VClass			Student's notes			Virtual machine	
			Relations	Relationships			Teacher board	
				Prerequisites			Video conference	

There are multiple notations that can be used to create the DSL. Some of them use text for describing instances (e.g. logic predicates), while others use a graphical notation (e.g. UML). They can also be for generic (e.g. XML) or specific purposes (e.g. the CSS language to describe HTML styles). However, in some cases, it is preferable to define a new and specific notation instead of using an existing one, because it is rare to find a specific existing notation that suits the model requirements (Kurtev et al., 2006). There are many generic notations available to use, but for delimited and relatively small domains, a specific notation is better than a generic one. On the one hand, it simplifies the modelling work and increases the level of expressiveness and precision (Kurtev et al., 2006). On the other hand, a specific notation reduces the chance of errors by teachers because it uses language elements from the given domain.

The notation chosen for the *VClass* ontology is a graphical notation that represents the model elements by types of icons interconnected by lines that represent relationships. The graphical notation is complemented by describing, for each element, a set of specific properties. Table 1 shows some of the most relevant graphical language elements used and the modelled concepts they represent.

To make the representation, consumption and modification of the classroom information easier, several diagram types have been created in order to represent the information about a particular classroom from different points of view. This separation in different views enhances the visibility and reduces the complexity of the diagrams.

The diagrams created to represent the classroom information can be classified into two main groups: the first one describes the compulsory information of the elements related to Virtual Classrooms and their relationships and the second one describes hierarchies between elements. For example, the classroom diagram or the subject diagram is located in the first group. The diagrams belonging to the second group are those that show competences, practical activities and task hierarchies. Table 2 enumerates the created diagrams.

3.3 A set of transformation rules to create the specified Virtual Classrooms

Once the models of a Virtual Classroom are created and its specification stored in the domain ontology, the next step is to use such specifications to create the implementation of the Virtual Classroom. In order to do so, a set of transformation rules should be defined to translate the specifications in terms of the VLE that will contain the Virtual Classroom. In this work, the defined rules create the Virtual Classrooms in Moodle, but its adaptation to other learning management systems is straightforward.

Since the main classes and properties of the domain ontology closely match with Moodle's resources, the translation process has been easy to define. Table 3 summarises some of the translations made between the specifications of Virtual Classrooms, which are written in terms of the domain ontology, and Moodle's resources.

Table 2 Types of diagrams

Main diagram	In this diagram, the user can create and edit subjects and Virtual Classrooms, and set relationships between them.
Classroom diagram	Each classroom has its own view, where the user can edit the resources that compose the classroom. This view can be either directly edited by the user or created by a wizard.
Subject's diagram	Subjects have a specific view where their competences can be defined. Also, the practical activities associated with that competence and the tasks related with them are shown in this diagram.
Competence's diagram	This view allows the practical activities required to achieve a competence to be edited.
Practical activity's diagram	The list of tasks to be done during a practical activity can be edited in this view.
Task's diagram	Shows and allows resources needed during the execution of a particular task to be edited.
Competencies diagram Practical activities diagram Tasks diagram	These diagrams show the hierarchy of the corresponding elements in terms of prerequisites relations.

3.4 The integration of the previous elements into a DSL

The next step is to create a tool that, using the proposed graphical notation, allows teachers to easily create instances of the ontology. This tool checks whether the instances satisfy the integrity constraints of the domain and generates the result using the defined transformation rules.

The term DSL is used to define the integration of (1) a conceptual schema that allows domain knowledge to be represented, (2) a language that allows specifying the domain knowledge and (3) a set of rules that allow translating the specifications of the domain into an implementation. There are some tools to ease the definition of DSLs. One of them is the tool inside the *Eclipse Graphical Modeling Project* (http://www.eclipse.org/ modeling/ gmp/, accessed on 21 January 2014) (e.g. GML or Graphiti) or the Microsoft *Visual Studio DSL Tools* (now known as Microsoft Visual Studio 2010 Visualization & Modeling SDK; http://www.microsoft.com/download/en/ details.aspx? displaylang=en &id=23025, accessed on 21 January 2014). These tools enable the definition of the graphical elements of the DSL notation. They provide the graphical tools needed to easily create diagrams and allow adding model validations and restrictions. For the *VClass* tool development, *Visual Studio DSL Tools* was chosen. The resulting tool is integrated inside the *Visual Studio Integrated Development Environment* (IDE).

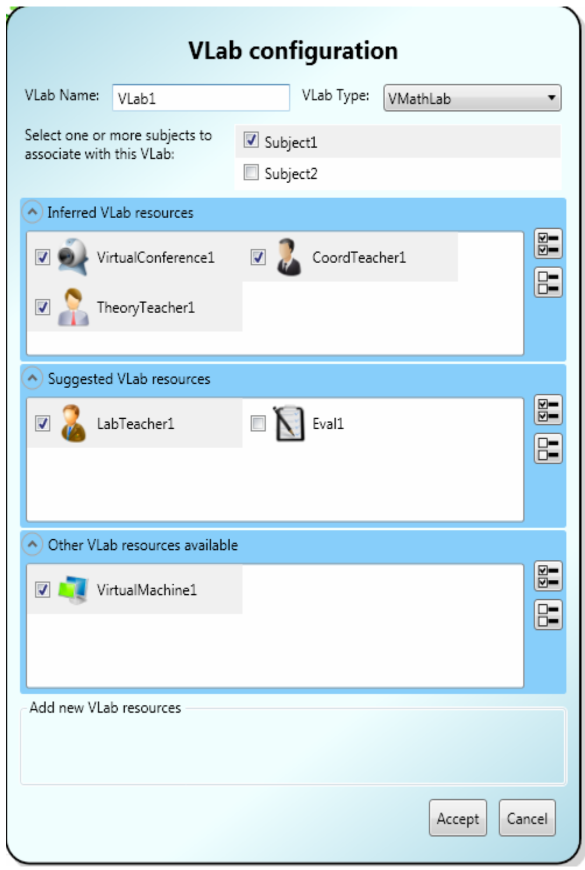
Table 3 Correspondence between Virtual Class elements and Moodle resources

VClass resource	*Attribute*	*Moodle resource*	*Attribute*
VClass	Name		
identifier			
creation date	Course	Full_name	
Course_ID			
Start_date			
Forum	VClass_ID		
name			
identifier	Forum	Course_ID	
name			
ID			
Wiki	VClass_ID		
name			
identifier	Wiki	Course_ID	
name			
ID			
FAQ			
Material			
Manual (guidelines)			
Software			
SoftwareSupport			
ComplementaryMaterial			
StudentNotes	VClass_ID		
ID			
name	Resource (file)	Course_ID	
ID			
name			
Simulator	VClass_ID		
ID			
name	Resource (html)	Course_ID	
ID			
name			

The resulting modelling tool provides different views to users for defining Virtual Classrooms, one for each kind of diagram described in the previous subsection. Each view or diagram of the Virtual Classrooms has a workspace where the graphical icons of the notation can be dropped, and the relationships between them established. The icon elements can be selected from a toolbar on the left and dragged and dropped onto the diagram. The types of relationships between elements may vary depending on the diagram view; they can also be selected from the toolbar and defined by drawing a line in the diagram between the two elements to be related. Besides the graphical visualisation, each element (icon) or relationship (line) has a set of associated properties, which can be edited in a form-like auxiliary window when the element is selected.

As previously stated, the *VClass* model itself does not directly guarantee that a Virtual Classroom contains all or part of the resources needed by the subjects associated with it. These resources are those needed by the tasks involved in practical activities, which are in turn related with the competences of the subjects associated with the virtual class (Vidal-Castro et al., 2012). In order to support the designer in the selection of the necessary resources for each classroom, a recommender system has been created (Figure 2). This recommender asks the user about the type of classroom and the subjects (from the model) that are associated with it. Then, it calculates the list of inferred resources (navigating the relationships from subject to resource passing through competences, practical activities and tasks hierarchy). A list of recommended resources is also suggested based on the type of VClass. For example, for programming a classroom the system proposes that it should have a compiler resource, while for a network classroom the system recommends a network simulator. As a final step, the wizard enables the user to choose other additional available resources or even to create a new one. After that, the tool automatically configures the chosen elements and relationships. Therefore, the recommender uses the VClass ontology information in order to enrich and complete the specification of Virtual Classrooms.

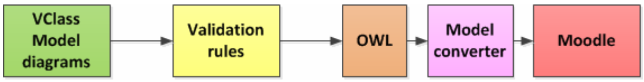
Figure 2 VClass configuration wizard (see online version for colours)



The tool also checks a set of validation rules that must be satisfied in order to ensure the accuracy of the model instantiation. The list of validation errors appears in a different window inside the same environment.

As Figure 3 shows, once every validation rule is satisfied, the tool stores the information of the different diagrams to the *VClass* ontology in OWL format (W3C, 2004). The defined *VClass* model is then used by a tool that physically generates the Virtual Classrooms using Moodle's php API, applying the transformation rules presented in the previous subsection.

Figure 3 VClass model transformation into physical Virtual Classrooms in Moodle (see online version for colours)



4 The case study

Even though the proposed system is generic enough to deal with any kind of Virtual Classroom, the case study proposed in this article is focused on the specific case of the Virtual Programming Laboratory (*VPLab*) at the UOC. Such laboratory supports the learning process of the practical programming activities and learners skills (Vallim et al., 2006).

The UOC is a fully online university that uses a Virtual Campus to teach in a VLE. It was created in 1994 in response to the new Information and Communications Technologies (ICT) reality and to fulfil the new learning needs of students.

From a technological perspective, the Virtual Campus allows an asynchronous and a synchronous communication between all UOC users: students, academic staff and managerial staff. Each subject is assigned a Virtual Classroom that contains all the necessary resources for the development of the learning process: mailboxes, blogs, wikis, videoconferences, public forums, agendas, a digital library, documentation, evaluation results and other e-learning-related tools. The Virtual Campus contains similar facilities to those that can be found at a conventional university campus, teaching, research, dissemination of knowledge and student services, but it still needs to improve the Virtual Laboratories.

The fundamental concepts of procedural programming are an essential part of Software Engineering and Computer Engineering, so Virtual Universities require *VPLabs*. Fundamental concepts include data types, control structures, functions, arrays, design of algorithms and the mechanics of running, testing and debugging (Cao et al., 2002). Furthermore, students must learn how to use at least one programming language.

4.1 Virtual laboratories framework

According to computer curricula (The Joint Task Force on Computing Curricula, 2004) and a review from ICECE07 (see Tovar and Castro, 2007) and networking curricula, practical laboratory activities are an essential part of any computer curriculum since they strengthen concepts presented during lectures. For this reason, new virtual resources are required in a Virtual University so that practical activities can be carried out. Such spaces are called Virtual Laboratories (Virtual Lab), and they should include different kinds of activities ranging from designing and implementing solutions to testing and documenting several applications, systems and processes. Virtual Laboratories are an indispensable resource for developing practical activities and skills in VLEs.

The term Virtual Lab is defined in different ways in the literature. Some authors (Chiu, 1999) take a simple vision of a Virtual Lab as a local computer hosting that may include some simulation (Harms, 2000; Leitner and Cane, 2005). An accurate definition can be found in Noor and Wasfy (2001), where a Virtual Lab is defined as a *leverage modelling, simulation, and information technologies to create an immersive, highly interactive virtual environment tailored to the needs of researchers and learners*. Other authors introduce other important aspects like pedagogical and academic factors into a Virtual Lab (Levert and Pierre, 2003; Rak and Godziemba-Maliszewski, 2006). Taking into account these established definitions, we define a Virtual Laboratory as an interactive virtual space that incorporates all the technological, pedagogical and human resources required for carrying out practical activities, which are then adapted to the needs of the students and teachers in a VLE. Therefore, they can be seen as a specialisation of a Virtual Classroom.

Apart from the VPLab, aforementioned, other different types of Virtual Labs have been proposed for other specific purposes. The most important in the fields of software engineering and computer engineering are the *Virtual Operating System Laboratory* (VOSLab), which allows students to carry out exercises involving operating systems manipulation and implementation, and the Virtual Networking Laboratory (VNLab), which allows the designing and programming of networking devices like routers and switches, and the programming of networking communication protocols. Other defined types of VLabs are the *Virtual Database Laboratory* (VDBLab), the *Virtual Computing Architecture and Organization Laboratory* (VCAOLab) and the *Virtual Mathematic Laboratory* (VMathLab).

Since 1997, in the engineering programs at the UOC, 73 different VLabs have been implemented (as of Spring 2013). Those VLabs are distributed as follows: 33 VPLabs, 12 VNLabs, 11 VMathLabs, four VOSLabs, 11 VDBLabs and two VCAOLabs. The following key resources were identified in the general structure of VPLabs:

1 Technological resources:

   a Virtual Communication Environment (VCE)

   b Simulator (SIM)

   c Remote Laboratory (REM)

   d Virtual Machine (VM)

   e Automatic Assessment Tool (AAT)

2 Pedagogic and strategic resources:

  f Learning Methodology (MET)

  g Support Documentation and Other Materials (DOC)

  h Evaluation (EVA)

3 Academic staff resources:

  i Teacher (TCH).

An overview of these resources is presented in Prieto-Blázquez et al. (2008).

4.2 Virtual classrooms implementation at UOC

The case study presented in this paper is based on the use of the presented approach to design and create a virtual programming laboratory for the subject of 'Programming Fundamentals' on Moodle LMS. Even though the proposed approach can be used in any kind of Virtual Classroom, we decided to test it with virtual laboratories since they are a kind of classroom with a more complex structure.

This section continues with a brief explanation of the benefits of having a DSL to define Virtual Classrooms instead of having to create them manually. The following subsections explain the process of modelling, checking and finally generating a Virtual Classroom (a VPLab in our case) following the MDD approach presented in Section 3.

4.3 Benefits of using a DSL for Virtual Classroom design

There are basically two roles in the process of creating Virtual Classrooms: teachers and management staff. Teachers are responsible for defining the Virtual Classroom needs and for specifying the corresponding resources. It is the management staff's responsibility to create and maintain the virtual class and its resources.

Without a DSL, teachers have to do ad hoc virtual class specifications, either on paper or in a digital document, which can be hard to handle or reuse, and is prone to errors. A DSL approach provides a well-defined structure with easy-to-use graphical model and integrity error checking.

Moreover, the ad hoc way of defining Virtual Classrooms may be misunderstood by management staff, so the final result may not fit the original purposes. With a DSL-defined Virtual Classroom, the management staff have a clear-cut specification of the virtual class, which can even be automatically implemented into the LMS system, using the DSL tools, so they have less work and less chance of errors. In addition, it is hard for them to know whether there is a virtual class configuration already available that is suitable for their purposes. The use of DSL enhances reusability, so teachers can search an institutional repository of the previously created Virtual Classrooms, choose one and create a new one from it.

4.4 Modelling the virtual laboratory

The teacher, who is responsible for designing the Virtual Classroom, defines a *Virtual Laboratory* by following these five steps using the DSL tool:

1 The teacher creates a new model and inserts a new subject in it. Information about the course is also specified, like course name (*PF1*: *Programming Fundamentals*) and some optional descriptions, as shown in Figure 4.

2 The teacher selects the main competencies that should be acquired by the students. In this case study, they are C1, C2, C3, C4 and C5, as shown in Figure 5:

- *C1*: learn the reasons to solve problems by using algorithms;
- *C2*: define the basic properties of an algorithm;
- *C3*: identify useful data structures to represent specific types of information;
- *C4*: write and compile a program that uses each of the main data structures;
- *C5*: write, test and debug simple functions and procedures.

3 The DSL tool shows a list of practical activities (*PAC1*, *PAC2* and *PAC3*) that students have to do in order to achieve the competencies previously defined (see Figure 5).

4 Then, a list of tools is suggested to carry out each practical activity. Figure 6 shows the five tasks to complete for practical activity number one (*PAC1*). Figure 6 also displays that the *read documentation* task consists of reading a *manual of integrated development environment*, *complementary material* and *FAQ's compiler*. By default, all the tools of VPLab are checked.

5 The provided tool includes a wizard that simplifies the task of creating the VClass configuration and associates it with all necessary resources. Figure 7 shows a screenshot of this wizard where the user can select the name and type of the VClass and the subjects associated with it. The wizard automatically calculates the set of resources that are required for the selected subjects (through the defined relations from subject to competences → practical activities → tasks → resources), and suggests other possible resources suitable to the type of VClass selected. At the end, the user can select other existing resources.

Figure 4 Subject's properties

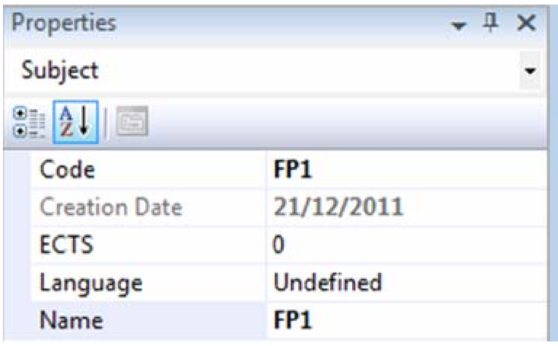


Figure 5 Selection of competencies (step 2) and selection of practical activities (step 3) (see online version for colours)

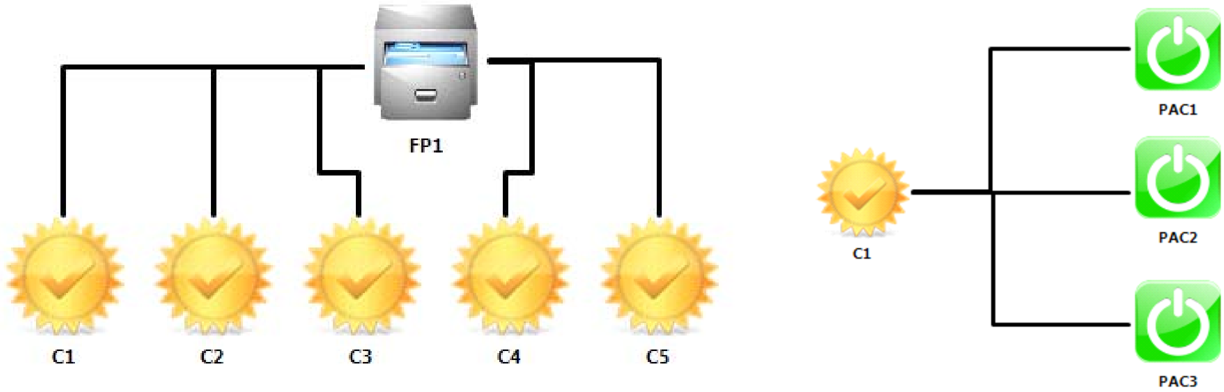


Figure 6 Tasks required in a practical activity and the resources used in such tasks (step 4)

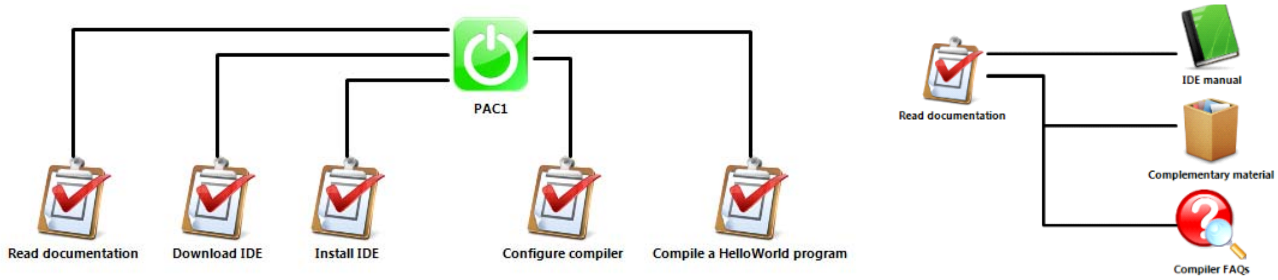
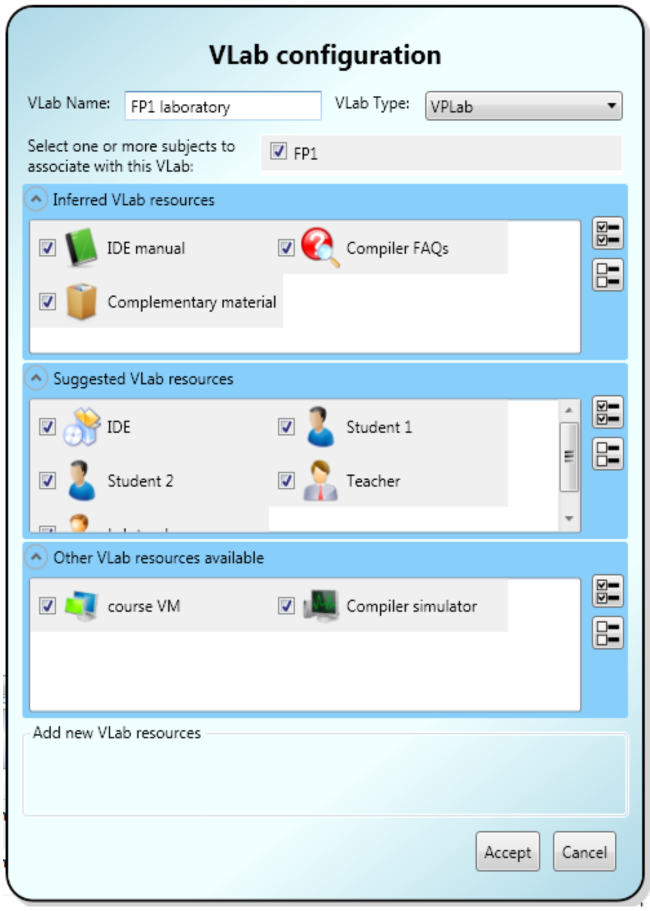


Figure 7 VPLab wizard (see online version for colours)



4.5 *Checking the virtual laboratory*

The DSL tool checks the design for inconsistencies and shows a list of errors that must be corrected in order to satisfy the integrity constraints of the VClass model. For instance, these integrity constraints include the specification of some compulsory metadata like the name of the model elements or the type of the VClass. In addition, there are some relationships that should be completed in order to obtain a consistent model; for example, a subject must define some competencies and a competency must define some practical activities, etc. There are other more specific constraints, such as the following: at minimum, a theory teacher and a coordinator must be specified for a *Subject*, or a practical activity must be associated with a corrector. Finally, the tool provides a set of warnings or recommendations about potential missing relationships, like the association between a laboratory and a specific type of resource. Lastly, the tool suggests the best type of lab to use.

4.6 *Implementing the virtual laboratory automatically from its specification*

When the teacher validates the final structure of the Virtual Classroom, the DSL tool creates a Moodle Classroom that includes all the necessary tools and properties previously chosen. Figure 9 shows the real Moodle Virtual Classroom's of *Programming Fundamentals*. Moodle Virtual Classrooms can be applied to any VLE. In our case, this Moodle Classroom has been integrated in the UOC Virtual Campus, because it is the default LMS used by UOC students.

Figure 8 Validation errors and warnings (see online version for colours)

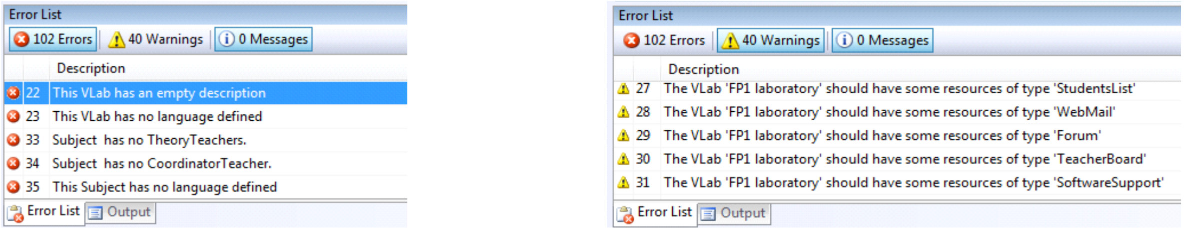


Figure 9 Moodle classroom (see online version for colours)

5 Conclusions

The main contribution of this paper is the creation of an MDD approach that reuses existent domain ontologies. This system is a DSL for describing Virtual Classrooms graphically and generating their implementation in VLEs. The proposed DSL eases the implementation of the classrooms in different VLEs and requires only domain knowledge to use it. It has been tested in a real environment: the creation of the virtual laboratory at the UOC.

As previously mentioned, some of the benefits of using the proposed DSL are (a) quality, through the reduction of errors in the creation and management of Virtual Classrooms; (b) reusability, through the facility to create new Virtual Classrooms adapted to different VLEs and semesters using previous Virtual Classrooms; (c) validity, through the use of the relevant concepts and their interrelations defined by domain experts and (d) feasibility, through the possibility that the Virtual Classroom can be built from its specification.

From our point of view, after the lessons learnt during the case study, the main improvements to be made are related to three facets of the DSL: the domain of interest, the user interaction and the functionalities covered.

The improvements related with the domain of interest refer to the evolution of the presented system to also deal with learning processes and training pathways. So we will provide a graphical environment to model and implement the place where the learning happens (Virtual Classrooms), the educational process that occurs (learning process) and what learning activities each student should perform (training paths). From the point of view of the user interaction, some improvements could also be done to the DSL, e.g. creating some new wizards to simplify tasks or automatically correct part of the inconsistencies and warnings detected by the tool. It will also be considered that other functionalities are provided, such as the reporting or the use of reverse engineering techniques to automatically generate Virtual Classroom diagrams from existing Virtual Classrooms. The integration of these improvements will facilitate the reuse of learning processes and resources in different semesters, VLEs and universities.

Acknowledgements

This paper has been partially supported by the Spanish Ministry of Science and Innovation under projects 'CO-PRIVACY' (TIN2011-27076-C03-02), CONSOLIDER INGENIO 2010 CSD2007-0004 'ARES', 'Design and construction of a Conceptual Modeling Assistant' (TIN2008-00444/TIN) and 'MAVSEL' (TIN2010-21715-C02-02).

References

Abdallah, F., Toffolon, C. and Warin, B. (2008) 'Models transformation to implement a Project-Based Collaborative Learning (PBCL) scenario: Moodle case study', *Proceedings of the Eighth IEEE International Conference on Advanced Learning Technologies*, 1–5 July, Santander, Cantabria, pp.639–643.

Amorin, R., Sánchez, E. and Lama, M. (2006) 'The problem of LD execution and the need for an IMS LD level B ontology', Nejdl, W. and Tochtermann, K. (Eds): *Proceedings of the First European Conference on Technology Enhanced Learning*, 1–4 October, Crete, Greece, *LNCS*, Vol. 4227, pp.537–542.

Babic, F., Wagner, J. and Paralic, J. (2008) 'The role of ontologies in collaborative systems', *Proceedings of the IEEE 6th International Symposium on Applied Machine Intelligence and Informatics*, 21–22 January, Herlany, pp.119–124.

Berlanga, A. and Garcia, F.J. (2005) 'Learning technology specifications: semantic objects for adaptive learning environments', *International Journal of Learning Technology*, Vol. 1, No. 4, pp.458–472.

Cao, J., Chan, A., Cao, W. and Yeung, C. (2002) 'Virtual programming lab for online distance learning', *Proceedings of the First International Conference on Advances in Web-Based Learning*, 17–19 August, Hong Kong, pp.216–227.

Chiu, K. (1999) 'What are the benefits of a virtual laboratory for student learning?' *Proceedings of the HERDSA Annual International Conference*, 12–15 July, Melbourne, pp.12–15.

Dodds, P. and Thropp, S.E. (2006) *Sharable Content Object Reference Model (SCORM) 2004, Version 1.0*, 3rd ed., Advanced Distributed Learning, Alexandria, VA.

Dodero, J.M., Martinez del Val, A. and Torres, J. (2010) 'An extensible approach to visually editing adaptive learning activities and designs based on services', *Journal of Visual Languages and Computing*, Vol. 21, No. 6, pp.332–346.

Dodero, J.M., Sánchez-Alonso, S. and Frosch-Wilke, D. (2007) 'Generative instructional engineering of competence development programmes', *Journal of Universal Computer Science*, Vol. 13, No. 9, pp.1213–1233.

Dodero, J.M., Ruiz-rube, I., Palomo-Duarte, M. and Cabot, J. (2012) 'Model-driven learning design', *Journal of Research and Practice in Information Technology*, Vol. 44, No. 3, pp.61–82.

Garcia Barriocanal, E., Sicília Urban, M.A. and Sánchez Alonso, S. (2012) 'Computing with competencies: modelling organizational capacities', *Expert Systems with Applications*, Vol. 39, No. 16, pp.12310–12318.

Ghebghoub, O., Abel, M.H. and Moulin, C. (2008) 'Semantic indexing of e-learning resources', *Proceedings of the 3rd International Conference on Information & Communication Technologies: From Theory to Applications (ICTTA 08)*, 7–11 April, Damascus, pp.1–6.

Ghebghoub, O., Abel, M.H., Moulin, C. and Leblanc, A. (2010) 'Building and use of a LOM ontology', in Gargouri, F. and Jaziri, W. (Eds): *Ontology Theory, Management and Design: Advanced Tools and Models*, IGI Global, Hershey, PA, USA, pp.162–178.

Harms, U. (2000) 'Virtual and remote labs in physics education', Proceedings of the *2nd European Conference on Physics Teaching in Engineering Education*, 14–17 June, Budapest, Hungary.

IEEE (2002) *Learning object metadata specification*. Available online at: http://ltsc.ieee.org/wg12/ (accessed on 21 January 2014).

IMS Global Learning Consortium, Inc. (n.d.) *IMS Learning Design Specification*. Available online at: http://www.imsglobal.org/learningdesign/ (accessed on 21 January 2014).

Jovanovic, J., Gasevic, D., Brooks, C.A., Knight, C., Richards, G. and Mccalla, G.I. (2006) 'Ontologies to support learning design context', *Lecture Notes in Computer Science*, Vol. 4227, pp.615–620.

Kurtev, I., Bézivin, J., Jouault, F. and Valduriez, P. (2006) 'Model-based DSL frameworks', *Proceedings of the Companion to the 21st ACM SIGPLAN Symposium on Object-Oriented Programming Systems, Languages, and Applications*, 22–26 October, Portland, OR, pp.602–161.

Leitner, L.J. and Cane, J.W. (2005) 'A virtual laboratory environment for online it education' *Proceedings of the 6th Conference on Information Technology Education*, 20–22 October, Newark, NJ, USA, pp.283–289.

Levert, C. and Pierre, S. (2003) 'Designing distributed virtual laboratories: methodological and telecommunications aspects', *International Journal on E-Learning*, Vol. 2, No. 3, pp.18–28.

MDA (2003) *Technical Guide to Model Driven Architecture: The MDA Guide v1.0.1*, Object Management Group, Needham, MA.

Noor, A.K. and Wasfy, T.M. (2001) 'Simulation of physical experiments in immersive virtual environments', *Engineering Computations: International Journal for Computer-Aided Engineering and Software*, Vol. 18, Nos. 3–4, pp.515–538.

OMG (2013) *MDA success stories*. Available online at: http://www.omg.org/mda/products_success.htm (accessed on 21 January 2014).

Pahl, C. and Melia, M. (2006) 'Semantic modelling of learning objects and instruction', *Proceeding of the 1st European Conference on Technology Enhanced Learning (EC-TEL 2006)*, 1–4 October, Crete, Greece, *LNCS*, Vol. 4227, pp.512–517.

Prieto-Blázquez, J., Garcia-Torà, I., Herrera-Joacomartí, J. and Guerrero-Roldán, A. (2008) 'Virtual laboratory ontology for engineering education', *Proceedings of the 38th Annual Conference on Frontiers in Education*, 22–25 October, Saratoga Springs, NY, pp.S2F-1–S2F-6.

Rak, A.M. and Godziemba-Maliszewski, M. (2006) 'A proposal of virtual laboratory structure', *Proceedings of the 23rd IEEE Instrumentation and Measurement Technology Conference*, 24–27 April, Sorrento, Italy, pp.847–850.

Rius, A., Conesa, J. and Gañán, G. (2010) 'A DSL tool to assist specifications of educational settings', *Proceedings of the International Conference on Semantic Web and Web Services*, 12–15 July, Las Vegas, NV, pp.12–118.

Rius, A., Conesa, J., García-Barriocanal, E. and Sicilia, M.A. (2013) 'Specifying patterns of educational settings by means of ontologies', *Journal of Universal Computer Science*, Vol. 19, No. 3, pp.353–382.

Rius, A., Conesa, J., García-Barriocanal, E. and Sicilia, M.A. (2014) 'Automating educational processes implementation by means of an ontological framework', *Computer Standards & Interfaces*, Vol. 36, No. 2, pp.335–348.

Rodríguez, M.E., Conesa, J. and Sicilia, M.A. (2009) 'Clarifying the semantics of relationships between learning objects', *Proceedings of 3rd International Conference on Metadata and Semantic Research*, 1–2 October, Milan, Italy, pp.35–47.

Sánchez-Alonso, S., Sicilia, M.A. and Pareja, M. (2007) 'Mapping IEEE LOM to WSML: an ontology of learning objects', *Proceedings of the 2nd International Conference on Internet Technologies and Applications*, 4–7 September, North East Wales Institute of Higher Education (NEWI), UK, pp.92–101.

The Joint Task Force on Computing Curricula (2004) *Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering*, IEEE Computer Society Press/ACM Press, Washington, DC/New York.

Tovar, E. and Castro, M. (2007) 'Building common spaces in engineering education: a review from ICECE05', *IEEE Transactions on Education*, Vol. 50, pp.79–84.

Vallim, M., Farines, J. and Cury, J. (2006) 'Practicing engineering in a freshman introductory course', *IEEE Transactions on Education*, Vol. 49, No. 1, pp.74–79.

Vidal-Castro, C., Sicilia, M.A. and Prieto, M. (2012) 'Representing instructional design methods using ontologies and rules', *Knowledge Based Systems*, Vol. 33, pp.180–194.

W3C (2004) *OWL web ontology language reference*. Available online at: http://www.w3.org/TR/owl-ref/ (accessed 21 January 2014).

Wilson, R. (2004) *The Role of Ontologies in Teaching and Learning*, TechWatch Reports, JISC, Bristol, UK.