A model-driven approach to e-course management

Goran Savić and Milan Segedinac
Faculty of Technical Sciences, University of Novi Sad, Serbia

Dušica Milenković, Tamara Hrin and Mirjana Segedinac
Faculty of Sciences, University of Novi Sad, Serbia

This paper presents research on using a model-driven approach to the development and management of electronic courses. We propose a course management system which stores a course model represented as distinct machine-readable components containing domain knowledge of different course aspects. Based on this formally defined platform-independent source course model, the system programmatically generates a final course in different platform-specific target models. Currently, our system supports the generation of IMS learning design, SCORM, LAMS and Sakai courses. The case study presents a formal model of the Web programming course and its transformation to the supported target models.

Introduction

Technology-enhanced learning forces the educational community to deal with course management issues. In order to exploit the advantages of contemporary educational technologies, the community should provide software support for the development, storage and usage of courses in learning management systems (LMSs). The basic idea behind the research presented in this paper is applying the principles of model-driven software development to the development and management of e-courses.

Model-driven software development is currently one of the leading techniques in the software development field. In short, this technique relies on the development of domain models which describe system data and behaviour at different abstraction levels (Brambilla, Cabot, & Wimmer, 2012). The development process is focused mainly on the representation of the domain knowledge, rather than on algorithms and programming. The basic principles of a model-driven approach (MDA) are illustrated in Figure 1.

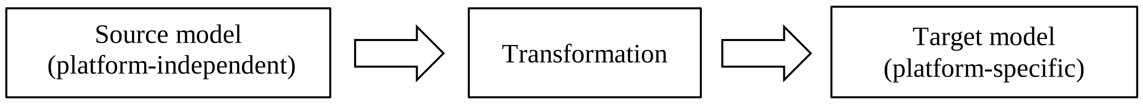

Figure 1. Overview of the MDA

The MDA starts with developing a source model. This source model is platform-independent and represents the domain knowledge without focusing on implementation details. Since it is an abstract representation, it cannot be directly used within any technological environment, but rather serves as an input point for further steps in the process of software development. In order to obtain a usable product from the source model, it must be transformed into a target model, which is a particular software application containing platform-specific source code. This process is usually automated. Model or code generators programmatically create platform-specific models or code based on a platform-independent model and a set of transformation rules. Presenting the platform-independent domain model separately from the final software application provides many benefits, such as portability (starting from the same domain model, final applications for various platforms can be generated), productivity (since the transformation process is usually automated, the MDA saves on resources by reducing the need for writing a platform-specific code manually) and maintainability (the availability of separately represented domain knowledge in the platform-independent manner gives domain experts direct access to the system specification).

This paper presents a software system which enables end users to exploit the benefits of the MDA in creating and managing e-courses. The system provides storage and management of e-courses, each represented by a platform-independent source model. Starting from such representation, the system programmatically transforms a course into different platform-specific formats. The course generated in such a format is a final product which is compliant with popular e-learning specifications and can be deployed on LMSs. Using the proposed course management system, we have separated course management, which is done in a platform-independent manner, from its usage within LMSs through a platform-specific target model.

The source model consists of distinct components defined in a machine-readable manner, where each component describes domain knowledge of a particular course aspect. We have developed the model on purpose to cover both technological and pedagogical aspects of a course. Namely, our model contains three basic components – learning objectives, learning resources and instructional strategy. Similar to code generation in model-driven software development, our solution uses this source platform-independent course model as an input for generating a course in different target platform-specific models. Currently, we have implemented transformation of our platform-independent model to four target platform-specific models. Those are models created within IMS learning design (IMS LD; IMS Global Learning Consortium, 2003) and SCORM (Advanced Distributed Learning, 2009) specifications, as well as models used by LAMS (http://www.lamsinternational.com) and Sakai (http://sakaiproject.org) software applications.

The rest of the paper is organised as follows. The next section presents current practices on modelling and management of e-courses and describes our motivation for the proposed research. The Course model section explains our course model and its components. Following that, we present a course management system which is based on this course model. Then, in the Case study section we detail a case study on modelling and management of one representative course. Finally, we present our conclusions and plans for further research.

Related work

There are two main approaches to course management. In the first approach, course management is integrated with course usage within the same software application. In the second one, course management is separate from the teaching process and is achieved through a separate software environment, producing a course that can be used by another software system, tailored for course usage in the teaching process.

An integrated approach has traditionally been used within LMSs. These systems provide creation, storage, management and usage of learning content (Ferris & Minielli, 2011). It is a common approach that an educational institution opts for a particular LMS which serves for the management as well as usage of all its courses. Such a vendor-centric approach to course management has some disadvantages. When choosing an LMS, it is necessary to consider a variety of aspects, making it difficult to decide on an LMS which best meets the requirements (Paulsen, 2003). Later migration to another LMS may be complicated (Jackson & D'Alessandro, 2004). Also, if an institution is large or decentralised, it is difficult to ensure the use of the same LMS and course format in all parts of the institution. Today's concept of open, globally accessible and shared courses calls for a course management approach which is more platform-independent (Duncan, 2003).

With regard to digital representations of a course, course models can be classified as content-centric or activity-centric (Blandin, 2004). Content-centric models recognise learning resources as basic course units. This model type considers a course as a repository of learning resources. The most popular content-centric model is the IMS content packaging (IMS CP) specification (IMS Global Learning Consortium, 2001). For activity-centric models, a basic course unit is a learning activity. A course is then defined as a process representing a workflow of learning activities. A learning process can be described formally using different languages, commonly named educational modelling languages (EML) (Rawlings, van Rosmalen, Koper, Rodríguez-Artacho, & Lefrere, 2002). Well-known models of this type are IMS LD and SCORM. These two models are not designed for any particular LMS, meaning that they can be used in any system which is IMS LD/SCORM-compliant. Among activity-centric models which are designed for a specific software platform, we should mention the model used within the LAMS system. A LAMS course consists of a sequence of learning activities. Beside these three popular activity-centric models, there has been much research on a formal representation of the learning process. Such formal descriptions are commonly named instructional design languages. Rodriguez, Derntl, and Botturi (2010) give an overview of these languages.

Regarding the expressiveness of different course models, content-centric models are often criticised for focusing solely on the technology, leaving the pedagogical aspect of e-learning aside (Hummel, Koper, & Tattersall, 2006). The pedagogical aspect covers the flow of the learning process, instructional design and other components that affect mastering the learning content. Activity-centric models may be considered more pedagogically expressive than content-centric ones. As explained by Karampiperis and Sampson (2005), to provide implementation of different pedagogical approaches, a course should be described as a set of learning activities.

Although activity-centric models are in general more expressive than content-centric models, they differ from one another in the level of pedagogical expressiveness. Among activity-centric models, the most pedagogically expressive one is the IMS LD model (Es & Koper, 2006). However, IMS LD shares one common disadvantage with other models – it has a monolithic structure. A monolithic course model does not separate course components into formally presented distinct units. Such a model makes course management less flexible since it is not possible to manage a single component independently. This complicates changing a course when we need to replace just one component (which is a common occurrence in practice).

The analysis presented in this section has shown that an efficient management of electronic courses should rely on a pedagogically expressive course model where course components are represented separately and can be managed independently from each other. In addition, course management should be independent from the system where the courses are used for teaching.

The course management system proposed in this paper tries to achieve this goal by using an MDA. In software development, the MDA ensures significant benefits (Kleppe, Warmer, & Bast, 2003), which we use here to improve current practices in course development. An MDA can improve productivity since it enables the generation of the final solution based on the source model. In the proposed course management system, we use the source course model to programmatically generate a target course that can be used in an LMS. Such an MDA ensures a platform-independent solution, which makes it more flexible for further changes in course formats and technologies. It is possible to support another target format, just by introducing a new set of transformation rules, with no changes in the source model and the system architecture. Another benefit of an MDA is that the existence of different domain models involves domain experts in the process of course development. Given that each course aspect is represented by a separate domain model, every expert can embed into a component knowledge that represents his/her field of expertise. The separately defined components contain domain knowledge in a machine-readable manner and may also be used for other scenarios which are not related to generating a course for an LMS. In addition, the course components are reusable in different courses. For example, the Case study section illustrates how the same learning content may be used within different instructional strategies.

The rest of this section presents similar research which has focused on building domain models for different course aspects. Bizonova, Ranc, and Drozdova (2007) use an MDA for representing an e-learning course. They propose a platform-independent course model which is transformed to two different platform-specific models. However, among all course aspects, the model is focused on learning resources only, resulting in a content-centric course model. Hernandez, Baldiris, Santos, Fabregat, and Boticario (2009) formally represent learning objectives as one of the components used for the automatic generation of the final course. The objectives are defined as an IMS reusable definition of competency or educational objective (RDCEO) competences. In contrast to the model of learning objectives proposed in this paper, the IMS RDCEO specification does not promote machine processing since most of the information is given in natural language. More expressive solutions for describing learning objectives have been used by Capuano, Gaeta, Micarelli, and Sangineto (2002) as well as Kontopoulos, Vrakas, Kokkoras, Bassiliades, and Vlahavas (2008) where objectives are formally represented using ontology. This approach is similar to the one taken in our research. However, in comparison with the latter two studies, our solution tries to give more attention to some pedagogical aspects of learning, meaning that we have built our ontology of learning objectives to rely on Bloom’s taxonomy (Bloom, Engelhart, Furst, Hill, & Krathwohl, 1956) as a classical paradigm. Capuano et al. (2002), Hernandez et al. (2009) and Kontopoulos et al. (2008) deal with learning resources in a course as well. Although Capuano et al. (2002) and Hernandez et al. (2009) do not provide any particular domain model for this purpose, the solution proposed by Kontopoulos et al. (2008) is quite similar to our model of learning resources. The system they proposed starts from raw learning content resulting in a programmatically generated IMS CP, which is a format also proposed by our model. Although the learning resources in the solution proposed by Capuano et al. (2002) are not organised in any particular structure, they are mapped to the ontology of learning objectives. The identical approach has been used in our research for describing relationships between learning resources and learning objectives. Other research which provides a formal representation of learning resources has been presented by Morales, Castillo, Fernandez-Olivares, and Gonzalez-Ferrer (2008). The resources are described using ontology, but there is no any explicit representation of the learning objectives. The relationships between the learning resources are described directly within the learning resources themselves. The domain models presented by Capuano et al. (2002), Kontopoulos et al. (2008) and Morales et al. (2008) are intended for the automatic generation of courses. As mentioned, the output format proposed by Kontopoulos et al. (2008) is the IMS CP, while the systems presented by Hernandez et al. (2009) and Morales et al. (2008) generate an IMS LD unit of learning. The solution proposed by Capuano et al. (2002) generates courses in a format specific for a learning platform developed within their research.

It should be noted that neither of the studies previously analysed in this section represents the instructional strategy of a course in a sufficient manner, which is one of the main features of our model. Hernandez et al. (2009) and Kontopoulos et al. (2008) use the planning domain definition language (PDDL) as a component for the course generation. This language defines a sequence of learning activities and may be considered a format for representing instructional strategy. However, PDDL has been designed for general planning, meaning that it is not sufficiently expressive in describing diverse instructional strategies. In addition, the studies are not focused on complete course management, but rather on its automatic generation only.

The system proposed in this paper tries to provide an integrated solution for the storage and management of different course components, as well as for the generation of target courses which are not limited to any single format. Also, the system architecture has been designed to promote later extension and support for other output formats.

Course model

In accordance with the analysis presented in the previous section, we have developed a course model which contains multiple components, each describing a particular course aspect. The course model is activity-centric and the components are represented as distinct units in a machine-readable format. We have tried to identify key components of an e-course which describe both technological and pedagogical issues. Since our model will be transformed to different target models, it must not be less expressive than any of these models.

The first step while developing the model was to identify key components of an e-course. To provide the desired expressiveness, we had to consider a wide set of course attributes. A course has often been identified with its learning resources, but it actually represents a much wider set of components (Koper, 2001). Tyrrell (1974) explains that the curriculum development process consists of:

- selection of learning objectives
- selection of learning experiences
- organisation of learning experiences
- evaluation of learning results.

These statements are mainly related to the traditional face-to-face learning. Segedinac, Savić, and Konjovic (2010) explain how these classic principles can be applied to e-learning. Besides the learning environment, learning experiences refer to any learning activity in a course. In e-learning, the learning activity is always related to an interaction with a particular digital learning resource. The organisation of learning experiences depends on the instructional strategy used in the course. The evaluation of learning results, as a last item in Tyrell's (1974) principles, is not the subject of this research. We cover this aspect only in the context of varying knowledge evaluation instruments (e.g., tests, exercises) as a subtype of learning resources.

Therefore, our course model identifies three basic components: learning objectives, learning resources and instructional strategy.

Figure 2 presents the proposed model and shows how it fits into the MDA of course development.

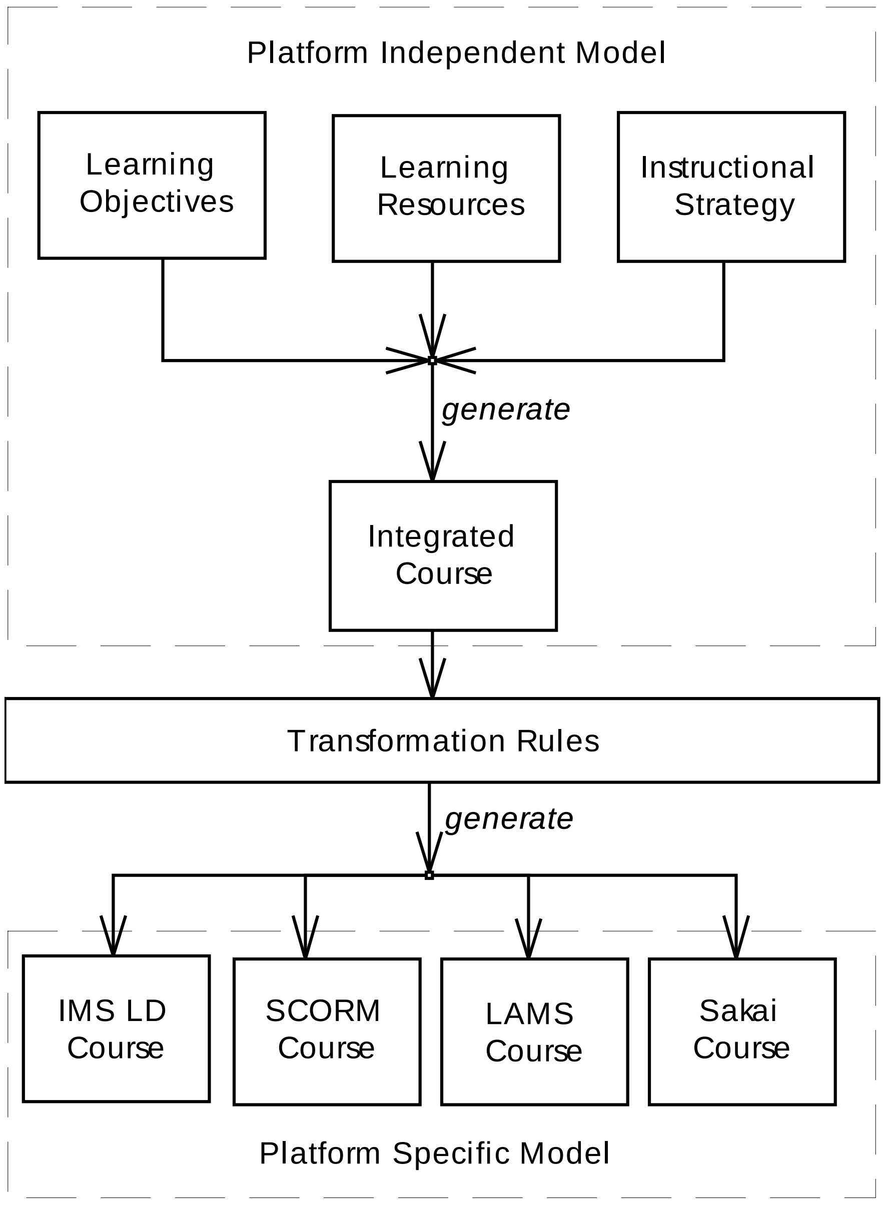

Figure 2. The proposed MDA for course development

The platform-independent model is shown at the top. It contains the three abovementioned basic components which are programmatically integrated into the fourth component. The fourth component represents a learning process which is shaped according to the specified instructional strategy. This learning process contains learning activities that use defined learning resources which are intended to achieve the specified learning objectives. This model of the integrated course is platform-independent too. By applying a set of transformation rules, the model of an integrated course is transformed to different target platform-specific models; we currently support four models shown at the bottom of the figure. Courses generated in one of these target models can be used in LMSs.

The following subsections present the components of our platform-independent model.

Learning objectives

The selection of learning objectives includes their specification, classification and organisation (Segedinac, Segedinac, Konjovic, & Savić, 2011). The specification of learning objectives refers to the representation of domain knowledge that should be mastered during the course. With regard to the classification, this research is focused on learning objectives in the cognitive domain. In this domain, our model classifies learning objectives according to Bloom’s revised taxonomy (Anderson & Krathwohl, 2001). This taxonomy represents the cognitive domain in two dimensions: cognitive process dimension and knowledge dimension. The cognitive process dimension consists of six categories: remember, understand, apply, analyse, evaluate and create. Cognitive processes which are categorised using this dimension are the result of the interaction of a learner with some learning content. The learning content belongs to some of the categories from the knowledge dimension. The categories in this dimension are factual knowledge, conceptual knowledge, procedural knowledge and metacognitive knowledge. The organisation of learning objectives is established by forming relations among them. Since learning objectives are hierarchically organised, a relation that defines the parent-child relationship between two objectives is one relation type that defines the structure of objectives. Beside this hierarchy, the structure is defined by the order of the objectives. Sometimes, a learner cannot master learning objectives in an arbitrary order since one objective can be a precondition for other objectives. Also, a learner may be advised to master one objective before another one, even if the first objective is not a strict precondition for the second one.

Once we have identified the necessary information on learning objectives, we have to choose an appropriate format for representing this information. For this purpose, we needed a structure which is sufficiently expressive to describe all the information, as well as to provide it in a machine-readable format. In accordance with this, we have chosen ontology as a formal way for representing our model of learning objectives. The classes defined in the ontology as well as important object properties are shown in Figure 3.

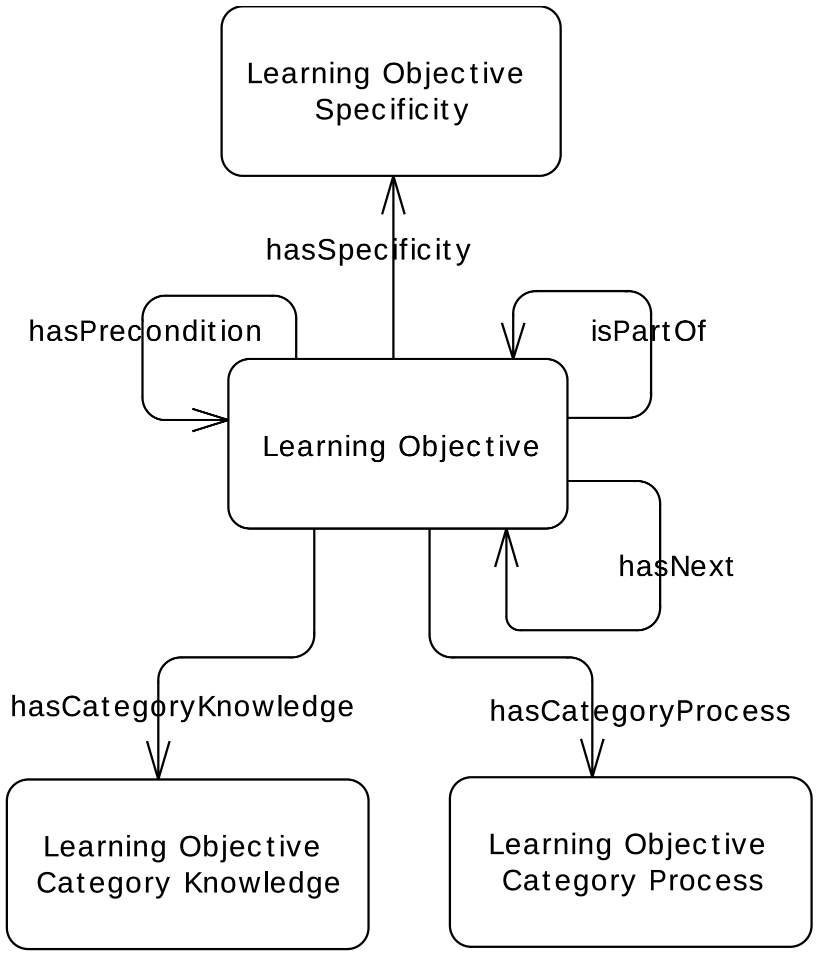
Learning Objective Specificity
hasSpecificity
hasPrecondition
isPartOf
Learning Objective
hasNext
hasCategoryKnowledge
hasCategoryProcess
Learning Objective Category Knowledge
Learning Objective Category Process

Figure 3. Ontology of learning objectives

Figure 3 shows that a learning objective is described by two dimensions from Bloom's revised taxonomy (Anderson & Krathwohl, 2001), as well as by its hierarchical level (represented by the *LearningObjectiveSpecificity* class). Object properties define relations between the learning objectives. The hierarchy among learning objectives is defined by the *isPartOf* property, preconditions by the *hasPrecondition* property, while suggested order can be established using the *hasNext* property.

Learning resources

In this research, under the term *learning resource* we consider any digital content that can be used for the achievement or evaluation of a learning objective in a course. With regard to formal representation of a learning resource, we recommend browser-readable formats to facilitate usage of the resource in Internet-based systems. Our model uses the IMS CP specification for formal description of the learning resources component. It has been chosen because it provides machine processing, it is widely adopted, and there are software tools for specifying learning resources in conformance with this specification. Our model relies on IMS CP built-in techniques for describing the semantics of learning resources using metadata. Just like IMS CP, our model is neutral with respect to the underlying metadata set.

Although learning objectives and learning resources are represented separately in our model, there is still a relationship between these two groups of information. For a specific learning resource, the model must contain information on the related learning objective. Similarly, we need to know which learning resources are used for the mastering or evaluation of the specific learning objective. This information is stored in our model using a separate component which maps learning resources to the ontology of learning objectives. Two mapping types have been defined:

- *contributes* – a learning resource contributes to the achievement of a learning objective
- *assesses* – a learning resource evaluates the achievement of a learning objective.

The separate mapping component ensures independent representation and management of different course aspects. In addition, since we map learning resources to the ontology of learning objectives, the ontology implicitly defines the relationships among learning resources. In this way, we avoid defining relations between resources directly by embedding this information into their content or by using a new structure. Such an approach facilitates changing relationships between resources as well as introducing new resources in a course. Just by mapping a resource to a specific learning objective, one can change or add to its relationship with other resources indirectly.

Figure 4 illustrates the mapping of learning resources to the ontology of learning objectives. For clarity, the relationships of learning objectives with the hierarchy level, the category of cognitive process and the category of knowledge are shown for one objective only (*Objective 1*).

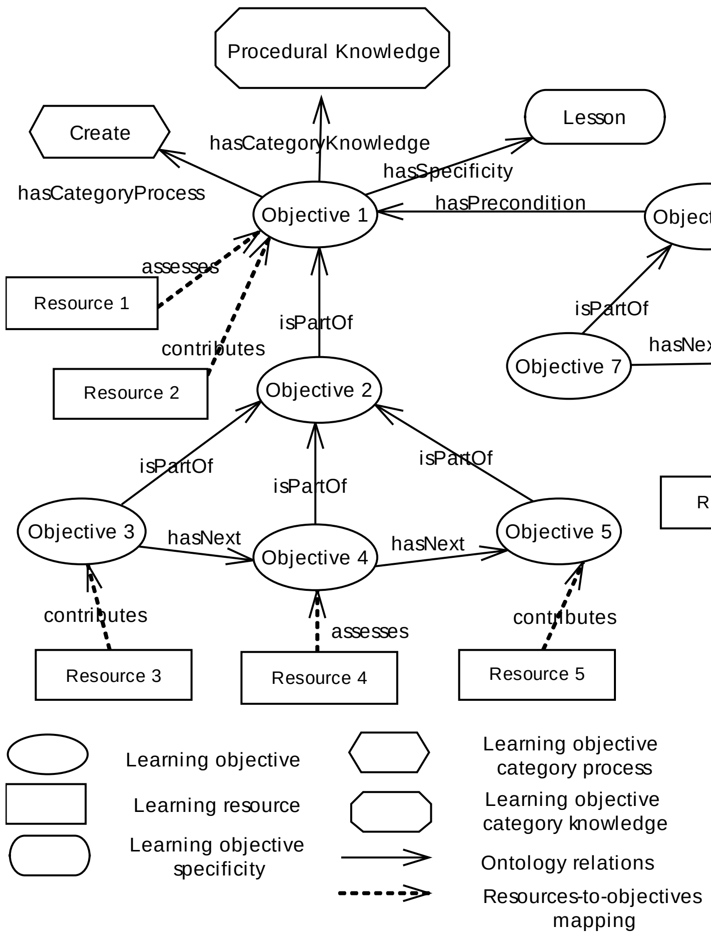

Figure 4. Mapping of learning resources to ontology of learning objectives

Just like the learning objectives component, the mapping component has also been represented using ontology. The corresponding eXtensible markup language (XML) schema is simple. It contains two classes, where the first one represents a learning objective and the second one a learning resource. Mapping between individuals of these two classes has been implemented using two object properties – *contributes* and *assesses*, which represent the abovementioned two types of relations between resources and objectives.

Instructional strategy

An instructional strategy in an e-learning course may be defined through a set of rules for the sequencing and selection of learning activities (Wiley, 2000). In the proposed model, this component represents a generic instructional strategy which is neutral to any specific course or learning content. A generic strategy, represented within this model component, can be applied to specific learning resources and objectives, defined within two other model components. This process is done programmatically, resulting in a specific course with a learning process shaped according to the specified instructional strategy.

As a machine-readable representation of generic instructional strategies, the model uses the e-learning instructional design meta-language (ELIDL) (Savić, Segedinac, Kovacevic, & Konjovic, 2013). ELIDL is an XML-based meta-language which provides an abstract definition of rules for the selection and organisation of learning activities, as well as the learning process workflow. The ELIDL XML schema and the description of the schema elements are publicly available in Savić (2015). In this paper, we will describe the ELIDL concepts through an example of one instructional strategy which is formally represented using ELIDL.

We will present a strategy named *remediation* whose description has been taken from Chew and Hua (2008). This strategy implies that a course contains an introduction, lessons and post-tests. After a learner has completed the lessons, post-tests are presented in order to retest the lessons. If the learner does not pass some post-test, he/she is directed to the corresponding lesson again.

The following text presents the ELIDL document that describes the remediation instructional strategy. Due to limited space, we will present only a part of the document containing a sequence of lessons, as well as the relation between lessons and post-tests.

Figure 5 presents a snippet of the ELIDL document which specifies a sequence of lessons in a course. For further relations, each lesson has the identifier *L*. The *selection-rule* tag specifies that a learner gets explanatory material first, then examples, and finally all other learning material from the lesson.

```
<sequence level="lesson" element-id="LS" sequence-element-id="L">
 <learning-object>
  <selection-rule>
   <include att-name="type" att-value="explanation-content"
            priority="1"/>
   <include att-name="type" att-value="example" priority="2"/>
   <include att-name="type" att-value="*" priority="3"/>
  </selection-rule>
 </learning-object>
</sequence>
```

Figure 5. A sequence of lessons for the remediation strategy represented using ELIDL

The relation between lessons and post-tests is shown in Figure 6. Among all lessons and post-tests, the relation is formed only between lessons and their corresponding post-tests. This is specified within the *join* tag. The *conditions* tag defines that the lessons related to the passed tests should not be available anymore.

```
<element-relationship>
 <element element-ref="L" alias="lesson"/>
 <element element-ref="POST_TEST" alias="test"/>
 <join>
  <equals>
  <name element="lesson"/>
  <parent-name element="test"/>
  </equals>
</join>
 <conditions>
 <if>
  <passed element="test"/>
 </if>
 <then>
  <hide element="lesson"/>
 </then>
 <else>
  <show element="lesson"/>
 </else>
 </conditions>
</element-relationship>
```

Figure 6. Lesson availability for the remediation strategy represented using ELIDL

Platform-independent integrated course

The integration of the three described model components results in the model of the integrated platform-independent course, which is the fourth source component of our model. Since our model is activity-centric, this component should formally describe the learning process as a sequence of learning activities. As mentioned, the most expressive activity-centric model is the IMS LD model. For that reason, this component works with concepts taken from the IMS LD specification. These concepts are formally represented using an in-memory structure described by Savić, Segedinac, and Konjovic (2011). The choice of IMS LD as a referent model has brought one more benefit. Since ELIDL has been designed as a meta-language for IMS LD, it is possible to directly map a generic ELIDL instructional strategy to an IMS LD-based learning process. In this context, an IMS LD-based learning process may be considered a concretisation of a generic ELIDL instructional strategy.

Course management system

This section presents a course management system built upon the model described in the previous section. Given that for each component the model contains a separate domain model formally represented in a different format, the system stores and manages each component independently. The system provides course management only, while the later usage of a course has been left to LMSs and other specialised software platforms. For that purpose, the system provides for exporting courses in different widely adopted formats.

A unified modeling language (UML) component diagram of the system architecture is shown in Figure 7.

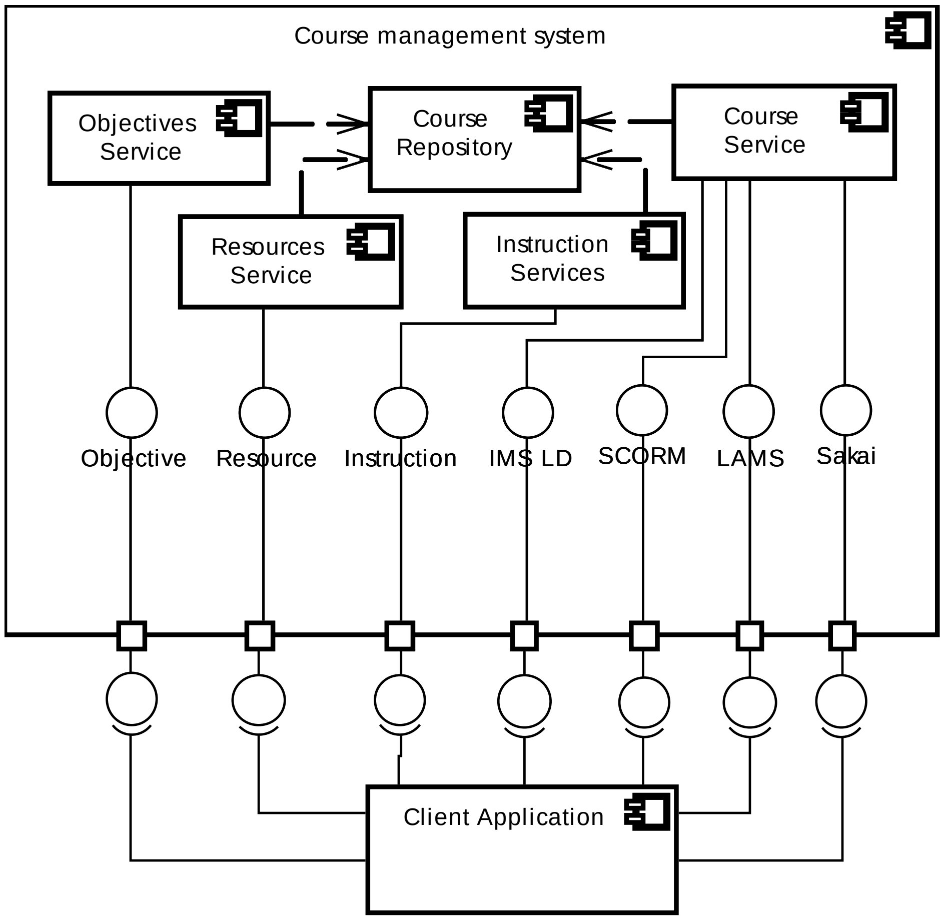

Figure 7. Course management system architecture

The system has been designed as a client-server application. The server part is represented by the *Course Management System* component in the figure. A core of the server part is the *Course Repository* component. This system component provides storage and access to the individual model components. Also, its responsibility is the generation of the integrated course in accordance with the model presented above.

The server interacts with the environment through a set of Web services. For each model component, there is a separate Web service which provides methods for adding and fetching a component to and from the system, respectively. The figure shows the corresponding Web services for the learning objectives, learning resources and instructional strategy model components. The *Course Service* system component provides target courses in different platform-specific formats. Currently, the system provides methods for generating and downloading a course in the IMS LD, SCORM, LAMS and Sakai course formats. In this way, our system may be considered knowledge-centric rather than document-centric. A course in this particular platform-specific format is just one of the possible representations of the course data stored within our platform-independent course model.

The system has been designed to facilitate later changes and extensions. For each model component, one can implement new service methods which would transform another format to the format expected by the proposed model. In addition, it is possible to support other platform-specific formats of target courses just by implementing an appropriate service method which would transform our integrated platform-independent course to a new target format. Therefore, since our system relies on an MDA, adding new features is done by implementing an appropriate set of transformation rules between the source and the target model.

The Web services provided are accessed by a client application represented by the *Client Application* component in Figure 7. The client application provides a graphical environment for accessing the application programming interface exposed by the server part of the system. Since Web services are platform-neutral, the client application can be implemented using different technologies. In this research, we have implemented it as a stand-alone application using .NET platform.

Case study

For the purpose of verifying the proposed course model, we have formally defined components of the Web programming course taught at the University of Novi Sad. Based on the defined source components, the proposed course management system has generated target Web programming courses in different formats.

According to the previously presented model of learning objectives, we have defined the ontology of learning objectives in the Web programming course. The classes related to the dimensions from the revised edition of Bloom's taxonomy (Anderson & Krathwohl, 2001) have predefined individuals which are the same for all courses. The individuals represent dimension categories. Similarly, there are predefined individuals for possible hierarchical levels represented by the *LearningObjectiveSpecificity* class. As a representation of learning objectives in the Web programming course, we have created 99 individuals of the *LearningObjective* class. Among these objectives, we have established relations of ordering and hierarchy using the corresponding object properties. This ontology of the Web programming learning objectives is publicly available in Savić (2015).

With regard to the learning resources component, the existing learning material of the Web programming course has been converted into HTML Web pages. The pages have then been organised into an IMS CP. The semantics of learning resources have been described using metadata conformant to the IEEE LOM specification (IEEE Standards Association, 2002). Besides learning resources, we have created an ontology that maps these resources to the previously defined ontology of learning objectives. The IMS CP and mapping file are both publicly available in Savić (2015).

As explained, the instructional strategy model represents a generic instructional strategy which can be applied to any learning content. For this case study, we have created six representative generic instructional strategies – *no-sequencing*, *linear*, *knowledge paced*, *competency assessment*, *remediation* (descriptions of the strategies are taken from Chew & Hua, 2008) and *project-based learning* (description taken from Derntl & Motschnig-Pitrik, 2004) – and described them using ELIDL. The corresponding ELIDL documents are publicly available in Savić (2015).

Defined course components have been added to the course management system using the client application and the corresponding Web services. Based on these input components, the system has generated platform-independent integrated courses. Since we have defined six different instructional strategies, we have six Web programming courses, where each course has been shaped according to a different instructional strategy.

Based on the platform-independent integrated Web programming courses, the system has generated the target courses in the IMS LD, SCORM, LAMS and Sakai formats.

With regard to the IMS LD as an output format, the system has generated six Web programming courses in the form of a IMS LD unit of learning. Given that the IMS LD model has significant pedagogical expressiveness, all six instructional strategies have been successfully represented within the IMS LD course. When it comes to learning objectives, the IMS LD specification represents a structure of learning objectives, but this structure is not sufficiently expressive to describe all the information defined in our model of learning objectives. For that reason, the system has used an alternative solution provided by the IMS LD specification. It has attached our learning objectives as an external file to the generated IMS LD unit of learning. With regard to the learning resources component, all the resources from the previously defined IMS CP have been imported programmatically into the generated IMS LD unit of learning.

The generation of the SCORM course has been implemented by the programmatic creation of a SCORM package with all the defined learning resources of the Web programming course. Since the SCORM specification does not provide an appropriate structure to represent the learning objectives from our model, the SCORM course does not contain information on learning objectives. The learning process workflow is defined by the SCORM specification as a sequence of learning activities. One can define simple conditions for moving from one learning activity to another. However, some of the more complicated nested conditions are not supported, which limits the flow control of a learning process represented by the SCORM specification. For that reason, it is not possible to fully represent our integrated course model using the SCORM specification. Among the six integrated Web programming courses, our system has generated a SCORM course only for a course created according to the *no-sequencing* instructional strategy. This strategy implies a relatively simple learning process workflow since a learner can access all the learning activities in an arbitrary order all the time.

LAMS courses are stored as XML documents. The proposed course management system has transformed a platform-independent representation of the Web programming course to a LAMS course. Since LAMS does not represent learning objectives, this component from our source model has not been represented in the target LAMS course. Among various LAMS tools, the only tool which enables working with universal LAMS-independent learning resources is *Shared Resources*. The tool displays different external learning resources to a learner. Our course management system has transformed the learning resources of the Web programming course to the LAMS *Shared Resources* learning activities. Just like our model, LAMS provides for defining a sequence of learning activities. However, the process workflow which is defined using our model cannot be represented in LAMS, because the LAMS workflow is controlled by internal LAMS-dependent resources. For that reason, it is not possible to map all the six platform-independent courses to the LAMS format. Similar to SCORM, the system has generated a LAMS course only for the Web programming course designed upon *no-sequencing* instructional strategy, which is the simplest strategy used in this case study. The LAMS course contains the learning activities of the Web programming course in the required order, but with no additional control of the learning process flow.

The Sakai course may contain different tools that interact with a learner. Since there is no appropriate tool for representing learning activities, we focused on representing learning resources which are provided by the *Resources* Sakai tool. Putting focus on the learning resources means that we consider the Sakai course as a content-centric model. For that reason, the proposed course management system has generated a Sakai course based on the learning resources component of our model only. The system has generated an archived Sakai course. For each Sakai tool, an archived course contains an XML document which represents the tool's content. Such an archived course can be imported into Sakai. Therefore, transformation of the platform-independent Web programming course to the Sakai course is achieved by generating an XML document that represents an archived file for the *Resources* Sakai tool. Also, all learning resources of the Web programming course are copied and together with the XML document packed into a zip file that contains the archived Web programming course. Since we consider the Sakai course as a content-centric format, the course that is generated does not represent the learning process in the course, meaning that none of the six instructional strategies have been built into the Sakai course. With regard to learning objectives, Sakai does not explicitly define them in a machine-readable format. The standard Sakai distribution contains a tool for describing course competences, but the description is given in natural language only, which prevents the representation of our model of learning objectives.

All the courses generated in the four above-mentioned formats are publicly available in Savić (2015).

If we analyse the transformation from our platform-independent model to the four supported platform-specific models, in some cases, a part of the expressiveness can be lost in the transformation. Some information from our source model cannot be represented in the target models. Neither of the analysed target models supports the structured description of learning objectives as defined by our source model. Learning resources from our model can be fully represented in any of the four target models. Based on an instructional strategy defined in a separate component of the source model, a learning process is generated as a sequence of learning activities.

This sequence can be mapped to all of the three activity-centric target models. However, sequencing rules defined by our generic instructional strategy can be fully represented in the IMS LD format only. In addition, the IMS LD format enables us to control the sequence flow with no changes in the learning resources themselves.

Conclusion

This paper details how a model-driven software development approach can be applied to the development and management of e-courses. Our research proposes a formal description of different aspects of an e-course through domain models. By integrating these input models, we programmatically generate a model of an integrated course, which represents a learning process in the course. This platform-independent model can be programmatically transformed to different target formats that are recognised by LMSs. The idea has been implemented within a course management system which stores separate course components and generates an integrated course in the following target models: IMS LD, SCORM, LAMS and Sakai.

Even though the proposed approach introduces some new opportunities for course representation and management, there are some constraints that may limit adopting the proposed solution into current educational settings. The MDA requires a significant amount of pre-processing. Namely, there is a need for developing domain-specific models, which in the long term produce a number of benefits, but in the short term hinder course development. Educational institutions are still mainly focused on the technological aspects of e-learning (i.e., publishing learning content online), while neglecting programmatic processing of domain knowledge established within classic educational research, which is provided by the solution proposed in this paper. Additionally, as stated at the end of the previous section, some parts of domain knowledge contained in the proposed model cannot be exploited by current LMSs nor represented by using popular specifications.

Further research will be conducted in two directions. The first direction is more technological and it is related to the extension of the current functionalities of the proposed course management system. We are planning to develop new services to support target course models used by Moodle and Canvas LMSs. The second direction of further research will try to extract separate course components from their implicit description within an integrated course. Accordingly, in comparison with the research presented in this paper, the transformation will be performed in the opposite direction. By taking a course in some of the popular target formats, we will try to extract an instructional strategy built into the course.

With regard to the adoption of the proposed approach in educational practice, the path for further research should follow the above-mentioned constraints. By developing appropriate software tools for representing domain knowledge in education, the time needed for course pre-processing can be reduced. Additionally, usage of the proposed approach within LMSs can be improved by developing software tools that can be easily integrated into them. Adopting the proposed solution could improve current educational practices in different ways. By being able to present their domain knowledge in technologically enhanced educational settings using the proposed model, teaching academics could bring together the technological and pedagogical aspects of their work more closely. Given that the model represents different aspects of domain knowledge separately, as independent components, the proposed solution fosters the involvement of different domain experts in course development, resulting in more comprehensive courses. In addition, separately presented components could be shared and reused more easily, which facilitates knowledge sharing, even among different subject areas. This can lead to different disciplines reusing the same instructional strategy.

Acknowledgments

Results presented in this paper are part of the research conducted within the Grant No. III-47003, Ministry of Education, Science and Technological Development of the Republic of Serbia.

References

Advanced Distributed Learning. (2009). Sharable content object reference model, SCORM 2004 (4th ed.). Retrieved from http://www.adlnet.gov

Anderson, L. W., & Krathwohl, D. R. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. New York, NY: Longman.

Bizonova, Z., Ranc, D., & Drozdova, M. (2007). Model driven e-learning platform integration. In K. Maillet, T. Klobucar, D. Gillet, & R. Klamma (Eds.), *Proceedings of the 2nd European Conference on Technology Enhanced Learning EC-TEL PROLEARN* (pp. 8–14). Retrieved from http://ceur-ws.org/Vol-288/p02.pdf

Blandin, B. (2004). Are e-learning standards neutral? In *Proceedings of the CALIE 04 International Conference on Computer Aided Learning in Engineering Education*. Retrieved from http://www-clips.imag.fr/calie04/actes/Blandin_final.pdf

Bloom, B. S., Engelhart, M. B., Furst, E. J., Hill, W. H., & Krathwohl, D. R. (1956). *Taxonomy of educational objectives: The classification of educational goals. Handbook 1: Cognitive domain*. New York, NY: Longmans Green.

Brambilla, M., Cabot, J., & Wimmer, M. (2012). *Model-driven software engineering in practice*. Williston, VT: Morgan & Claypool. doi:10.2200/S00441ED1V01Y201208SWE001

Capuano, N., Gaeta, M., Micarelli, A., & Sangineto, E. (2002). An integrated architecture for automatic course generation. In P. Kommers, V. Petrushin, Kinshuk, & I. Galeev (Eds.), *Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT 02)* (pp. 322–326). Retrieved from http://lttf.ieee.org/icalt2002/proceedings/

Chew, L. K., & Hua, T. G. (2008). Instructional strategies and limitations of the SCORM 2004 specification. In *Proceedings of the 16th International Conference on Computers in Education (ICCE 2008)* (pp. 153–160). Retrieved from https://pdfs.semanticscholar.org/eabd/45778eea0d0bc22ef381bbd3209f2cce09ca.pdf

Derntl, M., & Motschnig–Pitrik, R. (2004). Patterns for blended, person-centered learning: Strategy, concepts, experiences, and evaluation. In *Proceedings of the 2004 ACM Symposium on Applied Computing* (pp. 916–923). New York, NY: ACM Press. doi:10.1145/967900.968087

Duncan, C. (2003, February). *Digital repositories: E-learning for everyone*. Lecture presented at eLearnInternational, Edinburgh.

Es, R. V., & Koper, R. (2006). Testing the pedagogical expressiveness of IMS LD. *Journal of Educational Technology & Society*, *9*(1), 229–249. Retrieved from http://www.ifets.info/journals/9_1/19.pdf

Ferris, P. S., & Minielli, M. C. (2011). Using electronic courseware: Lessons for educators. In S. J. Hoffman (Ed.), *Teaching the humanities online: A practical guide to the virtual classroom* (pp. 26–45). New York, NY: M. E. Sharpe.

Hernandez, J., Baldiris, S., Santos, O. C., Fabregat, R., & Boticario, J. G. (2009). Conditional IMS learning design generation using user modeling and planning techniques. In I. Aedo, N. Chen, Kinshuk, D. Sampson, & L. Zaitseva (Eds.), *Proceedings of the 9th IEEE International Conference on Advanced Learning Technologies* (pp. 228–232). doi:10.1109/icalt.2009.185

Hummel, H., Koper, R., & Tattersall, C. (2006). LO -> LA: From a learning object centric view towards a learning activity perspective. *Technology, Instruction, Cognition, and Learning*, *3*(1), 15–18. Retrieved from http://www.academia.edu/1358012/LO-_LA_From_a_Learning_Object_centric_view_towards_a_Learning_Activity_perspective

IEEE Standards Association. (2002). IEEE standard for learning object metadata. Retrieved from https://standards.ieee.org/findstds/standard/1484.12.1-2002.html

IMS Global Learning Consortium. (2001). IMS content packaging information model version 1.1.2 final specification. Retrieved from http://www.imsglobal.org/content/packaging

IMS Global Learning Consortium. (2003). Learning design specification. Retrieved from http://www.imsglobal.org/learningdesign

Jackson, K., & D'Alessandro, N. (2004). Migrating to a new institution-wide learning management system: Challenges for staff development. In R. Atkinson, C. McBeath, D. Jonas-Dwyer, & R. Phillips (Eds), *Beyond the comfort zone: Proceedings of the 21st ASCILITE Conference* (pp. 458–467). Retrieved from http://www.ascilite.org/conferences/perth04/procs/pdf/jackson.pdf

Karampiperis, P., & Sampson, D. (2005). Designing learning services: From content-based to activity-based learning systems. In *Proceedings of the 14th International Conference on World Wide Web* (pp. 1110–1111). New York, NY: ACM. doi:10.1145/1062745.1062893

Kleppe, A. G., Warmer, J., & Bast, W. (2003). *MDA explained: The model driven architecture: Practice and promise*. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

Kontopoulos, E., Vrakas, D., Kokkoras, F., Bassiliades, N., & Vlahavas, I. (2008). An ontology-based planning system for e-course generation. *Expert Systems with Applications, 35*(1-2), 398–406. doi:10.1016/j.eswa.2007.07.034

Koper, R. (2001). Modeling units of study from a pedagogical perspective: The pedagogical meta-model behind EML. Retrieved from http://dspace.ou.nl/bitstream/1820/36/1/Pedagogical metamodel behind EMLv2.pdf

Morales, L., Castillo, L., Fernandez-Olivares, J., & Gonzalez-Ferrer, A. (2008). Automatic generation of user adapted learning designs: An AI-planning proposal. In *Proceedings of the 5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems* (pp. 324–328). Hanover: Springer-Verlag. doi:10.1007/978-3-540-70987-9_45

Paulsen, M. F. (2003). Experiences with learning management systems in 113 European institutions. *Educational Technology & Society*, *6*(4), 134–148. Retrieved from http://www.ifets.info/journals/6_4/13.pdf

Rawlings, A., van Rosmalen, P., Koper, R., Rodríguez-Artacho, M., & Lefrere, P. (2002). *Survey of educational modelling languages (EMLs)*. Retrieved from http://www.eife-l.org/publications/standards/elearning-standard/cenissslt/emlsurvey

Rodriguez, M. C., Derntl, M., & Botturi, L. (2010). Visual instructional design languages. *Journal of Visual Languages & Computing, 21*(6), 311–312. doi:10.1016/j.jvlc.2010.08.005

Savić, G. (2015). *Model-driven course management system – additional resources*. Novi Sad: University of Novi Sad, Chair of Informatics. Retrieved from http://informatika.ftn.uns.ac.rs/cormsh

Savić, G., Segedinac, M., & Konjovic, Z. (2011). The implementation of the IMS LD e-course generator. *e-Society Journal: Research and Applications, 2*(1), 121–131. Retrieved from http://www.tfzr.uns.ac.rs/esociety/issues/eSocietyVol2No1.pdf

Savić, G., Segedinac, M., Kovacevic, A., & Konjovic, Z. (2013). Measuring efficiency of formally represented instructional strategies. In *Proceedings of the 3rd International Conference on Information Society Technology and Management (ICIST 2013)* (pp. 274–279).Segedinac, M., Savić, G., & Konjovic, Z. (2010). Knowledge representation framework for curriculum development. In A. Fred & J. Filipe (Eds.), *Proceedings of the International Conference on Knowledge Engineering and Ontology Development* (pp. 327–330). doi:10.5220/0003052303270330

Segedinac, M. T., Segedinac, M. D., Konjovic, Z., & Savić, G. (2011). A formal approach to organization of educational objectives. *Psihologija 44*(4), 307–323. doi:10.2298/PSI1104307S

Tyrrell, R. (1974). An appraisal of the Tyler rationale. *The School Review, 83*(1), 151–162. doi:10.1086/443181

Wiley, D. (2000). *Learning object design and sequencing theory* (Doctoral dissertation). Brigham Young University, Provo, UT. Retrieved from https://opencontent.org/docs/dissertation.pdf

---

Corresponding author: Milan Segedinac, milansegedinac@uns.ac.rs

Please cite as: Savić, G., Segedinac, M., Milenković, D., Hrin, T., & Segedinac, M. (2018). *Australasian Journal of Educational Technology, 34*(1), 14-29. https://doi.org/10.14742/ajet.3124