Bridging the Gap between Practitioners and E-learning Standards: A Domain-specific Modeling Approach

Bridging the Gap between Practitioners and E-learning Standards: A Domain-specific Modeling Approach

Yongwu Miao, Tim Sodhi, Francis Brouns, Peter Sloep, and Rob Koper
Educational Technology Expertise Center, Open University of the Netherlands
E-mail: (yongwu.miao, tim.sodhi, francis.brouns, peter.sloep, rob.koper)@ou.nl

Abstract. Developing a learning design using IMS Learning Design (LD) is difficult for average practitioners because a high overhead of pedagogical knowledge and technical knowledge is required. Through using peer assessment as an exemplary pedagogy, this paper presents a domain-specific modeling (DSM) approach to a new generation of LD authoring tools, for enabling practitioners to create learning designs. Adopting a DSM approach, on the one hand, pedagogic experts develop a pedagogy-specific modeling language, in which notations are directly chosen from the concepts and rules used to describe pedagogic approaches. On the other hand, technical experts develop transformation algorithms, which will map the models represented in the pedagogy-specific modeling language into machine-interpretable code represented in LD. This technical approach to a new generation of LD authoring tools has been illustrated through presenting the whole procedure of the development of a peer assessment authoring tool.

Keywords: domain-specific modeling, IMS LD, IMS QTI, peer assessment

1 Introduction

IMS Learning Design specification (LD) [3] is a pedagogy-neutral and machine-interpretable educational modeling language. It can be used to describe a wide range of pedagogies as units of learning (UoL). However, developing a UoL using LD constructors (e.g., roles, learning activities, properties, and conditions) is not an easy task because the required level of pedagogical knowledge and technical knowledge is significant. Although several LD authoring tools have been developed, they assume a keen knowledge of the technical specification and thus are developed for experts, who can deal with pedagogic issues and handle technical complexity at the same time. Finding out how to empower practitioners, who cannot sustain a high overhead of pedagogical and technical knowledge, is crucial for the wide application of LD in practice. In this paper, we present a domain-specific modeling (DSM) approach to a new generation of LD authoring tools and show how it can help practitioners develop complicated learning designs without handling technical complexities of the open e-learning standards. Throughout the paper, we will use peer assessment as an exemplary pedagogy, although the DSM approach is in no way restricted to such an application.

2 Modeling Peer Assessment in IMS LD

Peer assessment is a process consisting of various cognitive activities such as reviewing, summarizing, clarifying, providing feedback, diagnosing errors, and identifying missing knowledge or deviations [10]. In the literature many peer assessment models have been described [4, 7, and 11]. Note that various peer assessment models are available in practice and there is no “one-size-fits-all”. The variables of the peer assessment could include levels of time on task, engagement, and practice, coupled with a greater sense of accountability and responsibility. Topping [8] developed a typology, which consists of a survey of variables found in the reported systems of peer assessment. These pedagogic issues have to be taken into account systematically for designing an effective and efficient peer assessment model.

A technical approach to script a peer assessment through a combined use of LD and IMS Question and Test Interoperability (QTI) [6] has been proposed in [5]. Various activities (e.g. designing assignment, writing report, reviewing, providing feedback, and identifying missing knowledge) performed by different learners (including the tutor) have to be modeled in sequence and/or in parallel as control-flow. Various information units (e.g., analysis reports and feedback, modeled as properties in LD) produced by using various services (e.g., text editor, QTI authoring tool and QTI player) and transferred between activities/peers have to be modeled as information flows. As indicated in [5], if the number of participants is large and the information exchange patterns are sophisticated, specifying a peer assessment model in terms of LD and QTI will be very complex and time-consuming.

3 Domain-specific Modeling

Domain-specific Modeling (DSM) or Domain-specific Modeling Language (DSML) is a new method in software development. It has been applied in many application domains. In comparison with the Unified Modeling Language (UML), DSM is more expressive and therefore tackles complexity better, making modeling easier [2]. In addition, DSM allows automatic, full code generation, similar to the way today's compilers generate Assembler from a programming language like JAVA [1].

DSM raises the level of abstraction beyond programming by specifying the solution in terms of concepts and associated rules extracted from the very domain of the problem being solved. The final software products are generated from this high-level abstraction [1]. Notations in a domain-specific model are a whole level of abstraction higher than those in UML. As shown in Figure 1, normally software developers will implement the final product by mapping the domain concepts to assembler, code, or UML model. By adopting the DSM, a meta-model of the problem domain will be constructed as a modeling language by domain experts. Domain-specific code generators and executable components will be developed by experienced technical experts. Hence, less experienced developers and even practitioners can understand, validate, and develop DSML programs through employing the concepts and rules familiar to them, whereas developing equivalent solutions in a general-purpose language such as UML or JAVA is often too daunting

a task for people typically not trained as software engineers. In addition it is often possible to validate and optimize at the level of the domain rather than at the level of general-purpose languages where detail may obfuscate important features [9].

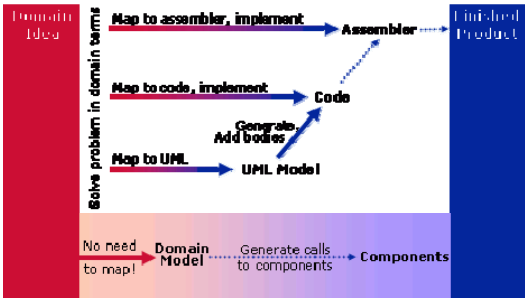

Figure 1: DSM and other software development approaches (taken from [1])

4 A Peer Assessment Modeling Language

The definition of a peer assessment modeling language should start with choosing the vocabularies used in the domain of peer assessment. Such vocabularies provide natural concepts that describe peer assessment in ways that practitioners already understand. They do not need to think of solutions in coding terms (e.g., classes, fields, and methods) or/and generic concepts (e.g., activities, action objects and decision points).

Based on the peer assessment models and the typology mentioned in the last section, we developed a peer assessment meta-model by deriving many of the modeling concepts and the constraints. As shown in Figure 2, a peer assessment process normally consists of four stages: *assessment design*, *evidence collection*, *giving feedback*, and *reacting to feedback*. In the assessment design stage, one or more various activities such as *constructing assessment form*, *designing assignment*, and *setting time* may take place. A *designer* can perform one or more activities and one activity can be done by one or more designers. Performing design activities may produce *assignment description* and/or *assessment forms*. Note that the assessment design stage may or may not be included in a peer assessment, because sometimes the assignment description and the assessment form have been pre-defined before the peer assessment starts. No matter whether the assessment design stage is included, a peer assessment actually starts from the evidence collection stage, in which one or more *candidates* do assignments such as *responding to questions* or *performing tasks* according to the assignment description. Then the *assignment outcomes* will be produced and distributed to the activities in a subsequent giving feedback stage, in which one or more *reviewers* will *comment on*, *rate*, and *grade* the allocated assignment outcomes using the assessment form, and finally provide feedback in forms of *comments*, *rates*, and *grades*. In summative peer assessments, the process may terminate here. In the formative peer assessment, typically a reacting to feedback stage will follow, in which the candidate may *view* or *review feedback*. Sometimes,

candidates further *improve* assignment outcomes and even *require elaborate feedback*. In the later case, the reviewer may *provide elaborate* or *additional feedback*. In some extreme situations, reacting to feedback stages and giving feedback can be repeated several times.

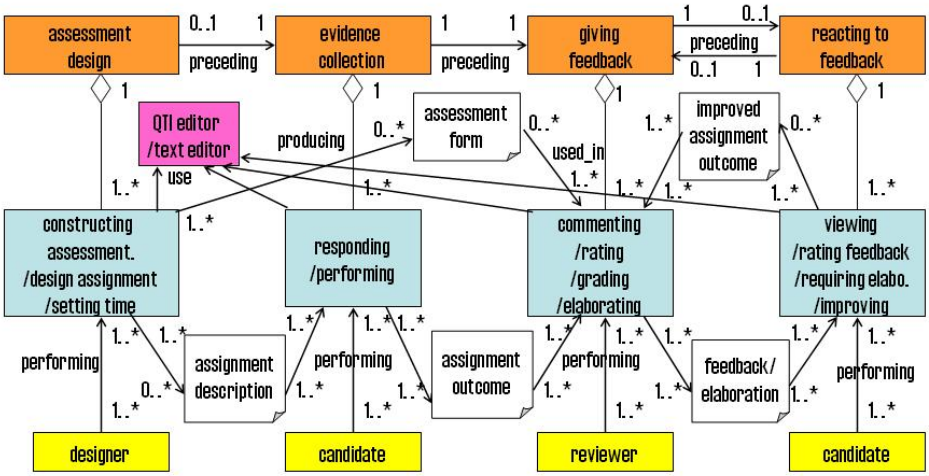

Figure 2: The meta-model of peer assessment

The peer assessment meta-model as just discussed is formally defined in XML schemas, which can be regarded as a high-level process modeling language to specify various peer assessment models. Note that the diagram of Figure 2 just illustrates the most important concepts of the meta-model and primary relationships between them. Many details of the modeling language are actually represented as alternatives, constraints, and rules, which have not been drawn in the diagram. When specifying a peer assessment model, one has to represent the design decisions in terms of the modeling language.

5 A Peer Assessment Authoring Tool

For experienced users, the peer assessment modeling language can be used to specific a peer assessment model directly in the form of XML. In order to support practitioners to develop online peer assessments, an authoring tool for modeling with the peer assessment modeling language should be provided.

Guidelines for design decisions: The peer assessment modeling language can be used to specify a peer assessment model directly. However, it would be nice if practitioners could be guided to make a series of design decisions. Figure 3 illustrates the design guidelines for developing a peer assessment model step by step. All design decisions will be captured, and then will be available for subsequent design and refinement in the process of modeling. In order to help practitioners make design

decisions, the peer assessment modeling language defines default values for certain design variables. For example, it is assumed that only two persons are involved in the process and both are *candidates* and *reviewers*. If the default values are not appropriate, practitioners can assign the variable values and thus customize the design. For example, it can be changed as five persons are engaged in the customized process and each reviews three of the others' assignment outcomes. Moreover, certain design decisions are related so that if one design decision has been made then another decision will be made accordingly. For example, if the purpose of the peer assessment to be modeled is a summative assessment, then the activity *improving assignment outcome* in the *reacting to feedback* stage and the activity *elaborating feedback* in the *giving feedback* stage will be excluded; there is no need then to specify them.

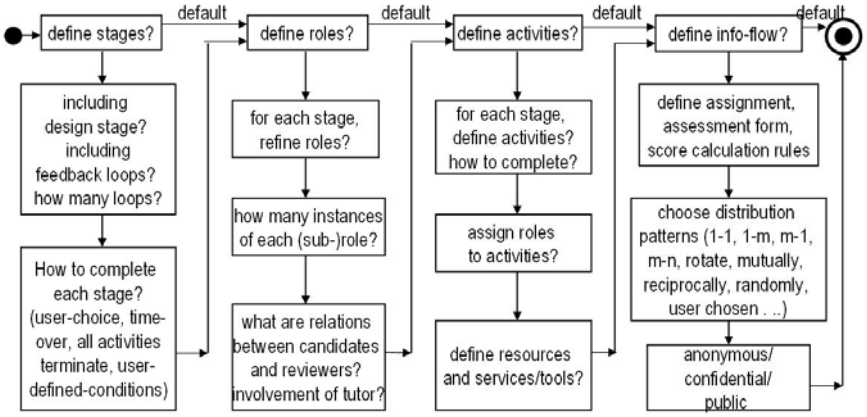

Figure 3: Design guidelines

User interface of the authoring tool: Based on the design guidelines described above, a 'wizard' is developed to guide the practitioner through a sequenced set of pages. The wizard page defines the controls that are used for making design decisions. It responds to events in its decision-making areas. After the practitioner has made choices or/and has provided input on the current page, he can go ahead by clicking the "Next" button. When required inputs from the practitioner on all relevant pages have been received, the wizard will complete and all decisions (including the default values selected) will be captured by the wizard and represented internally in the peer assessment modeling language.

Transforming algorithm: After a peer assessment model has been specified using the wizard, the authoring tool will transform the model, represented in the peer assessment modeling language, into an executable model, represented in LD and QTI. The basic idea of the transformation algorithm is to create a set of instances of domain-generic concepts for each instance of a domain-specific concept and to maintain their relationships. For example, the notation *commenting* in the peer assessment modeling language will be translated into a *support-activity* element of LD with an associated *environment* element that will be generated together with the *support-activity* element.

6 Conclusions

In this paper, we outlined an approach to apply the DSM paradigm to the development of a pedagogy-specific modeling language. Through developing a peer assessment authoring tool, we demonstrated that a DSM approach can be used to develop a new generation of LD authoring tools, for supporting practitioners to develop learning designs. Using such a pedagogy-specific modeling tool, practitioners can benefit from open technical e-learning standards without having to deal with their technical complexity. Moreover, the quality of the resulting models is higher on both pedagogical and technical aspects because experienced pedagogical and technical experts developed the domain-specific modeling language and the code generator. We will conduct experiments with the target user group after the tool is completely developed and tested. It is also expected that more pedagogy-specific modeling languages with different abstraction levels as a hierarchic structure and a corresponding authoring toolset will be designed and implemented in the future for practitioners to develop and combine various learning designs.

References

1. DSM forum: http://www.dsmforum.org/
2. Bruce, D. What makes a good domain-specific language? APOSTLE, and its approach to parallel discrete event simulation. In S. Kamin, editor, DSL'97 – First ACM SIGPLAN Workshop on Domain-Specific Languages, in Association with POPL'97, pages 17–35, Paris, France, January 1997. University of Illinois Computer Science Report.
3. LD: IMS Learning Design Specification. Available at: http://www.imsglobal.org/learningdesign/index.html (2003).
4. Liu, Z., Lin, S., and Yuan, S.: Experiencing NetPeas: Another way of learning. *Lecture notes in computer science*, vol. 2198, pp. 584-588 (2001).
5. Miao, Y. & Koper, R.: An Efficient and Flexible Technical Approach to Develop and Deliver Online Peer Assessment. In *Proceedings of CSCL 2007 conference*, p. 502-511, July 16-21, 2007, New Jersey, USA (2007).
6. QTI: IMS Question and Test Interoperability Specification. Available at: http://www.imsglobal.org/question/index.html (2006).
7. Sitthiworachart, J., and Joy, M.: Web-based Peer Assessment in Learning Computer Programming. In *Proc. of the 3rd IEEE ICALT03*, 180-184, July 9-11, 2003, Athens, Greece (2003).
8. Topping, K. J.: Peer assessment between students in colleges and universities. *Review of Educational Research*, 68: p. 249-276 (1998).
9. van Deursen, P. K. A. and Visser, J. Domain-specific languages: An annotated bibliography. *ACM SIGPLAN Notices*, 35(6):26–36, June 2000.
10. van Lehn, K. A., Chi, M. T. H., Baggett, W., & Murray, R. C.: *Progress report: Towards a theory of learning during tutoring.* Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh (1995).
11. Volder, M.D., Rutjens, M., Slootmaker, A., Kurvers, H., Bitter, M., Kappe, R., Roossink, H., Goeijen, J., and Reitzema, H.: Espace: *A New Web-Tool for Peer Assessment with In-Built Feedback Quality System.* in *Proceedings of ABR & TLC Conference*, 2007, Hawaii, USA (2007).