Knowledge-Based Systems 41 (2013) 26–42

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A Petri net model for changing units of learning in runtime

Juan C. Vidal *, Manuel Lama, Félix Díaz-Hermida, Alberto Bugarín

Centro de Investigación en Tecnoloxías da Información (CITIUS), Universidade de Santiago de Compostela, 15782 Santiago de Compostela, Spain

ARTICLE INFO

Article history:
Received 29 March 2012
Received in revised form 2 October 2012
Accepted 24 December 2012
Available online 3 January 2013

Keywords:
Runtime adaptation
Petri nets
Workflows
IMS Learning Design
Dynamic change

ABSTRACT

In this paper, we present a Petri net-based approach that facilitates making structural changes at runtime to units of learning specified in IMS Learning Design (IMS LD). The proposed change model makes use of the hierarchical Petri net model derived from IMS LD and a page substitution mechanisms to replace learning flow components on the fly. As a result, a new hierarchical Petri net is defined, verifying that changes do not cause inconsistencies, such as data loss or errors, in the learning process. Furthermore, two change modes have been implemented: a safe mode, for conservative modifications that do not affect participants, and an unsafe mode, which may require to transfer the participant execution to a safe point in the new structure. Our approach has been developed as an extension of a Petri net-based IMS LD engine and validated with a set of real units of learning.

1. Introduction

One of the objectives of next generation of technology enhanced learning is the definition of methods and models that facilitate *adaptive learning* [1]. Future systems will use the multiple sources of personalization from which they may create the *learning flow* appropriately, that is, adapting the coordination of the learning activities to be carried out by the participants in a course. Achieving this goal is difficult considering the many aspects that may influence this kind of learning [2]. In this sense, better adaptive learning models are needed for structuring the information to be taught, for tracking and learning about the student behavior, for designing the ways to convey the information, or for defining the user interaction with the system. Thus, the most used Educational Modeling Languages (EMLs) [3] for the specification of learning design, such as SCORM [4] or IMS[1] Learning Design (IMS LD) [5,6], have included some features for learning flow adaptation. At this respect, IMS LD has the advantage of providing the means to apply adaptation to groups of students, facilitating the development of collaborative learning, and not only personalized adaptation such as SCORM [7].

IMS LD is a metadata standard that describes the learning process [5] through three levels of implementation depending on whether the learning design is adaptive or not. The *Level A* defines the learning design, that is, the set of educational activities to perform, their coordination, the participants (usually students or instructors), and the resources used during the execution of the UoL. In this specification, a learning design can be considered as a learning workflow (or *learning flow*) that consists in the execution of a set of activities carried out by the participants in the UoL. However, it imposes a given structure to the learning flow, independent of the pedagogical methodology followed in the UoL but particularly suitable to collaborative learning. The *Level B* provides the means to adapt the learning design during the execution. It specifically adds *properties* and *conditions* that may enable the execution of activities. Finally, *Level C* incorporates notifications to *Level B*.

In last years, a number of tools have been developed for the execution of units of learning (UoLs) based on IMS LD [8–14]. These tools deal with the learning flow adaptation through the mechanisms specified in IMS LD, which consist basically in the activation of rules that show or hide activities or learning content [15]. This is a *dynamic adaptation* where all the paths of the learning flow are previously predefined in the design phase, and new activities or flow structures cannot be changed or introduced on the fly. However, the main objectives of this paper is to provide a model able to be changed on the fly, for example, by the instructors responsible for teaching the subject. This is a *runtime adaptation* that provides a great value to an e-learning system since it has a great capacity to adapt and respond to contingencies. Adaptive systems can indeed configure in runtime the most appropriate learning path for a student in terms of their ability or other criteria. However, when a course and its adaptive mechanisms have been defined, and this course is published, it is not possible to modify its

* Corresponding author. Tel.: +34 881816388.
E-mail addresses: juan.vidal@usc.es (J.C. Vidal), manuel.lama@usc.es (M. Lama), felix.diaz@usc.es (F. Díaz-Hermida), alberto.bugarin.diz@usc.es (A. Bugarín).

[1] IMS Global Learning Consortium is a nonprofit organization that strives to enable the growth and impact of learning technology in the education. The acronym IMS originally stood for Instructional Management Systems, but the full term was quickly dropped and now only IMS is used.

0950-7051/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2012.12.010

structure. So what happens when it is necessary to reinforce the learning of a particular subject, when new contents, that were not initially planned, must be added, or for example when the list of topics must be cropped? In such situations it is clear that the course should be redesigned and adapted to the characteristics of students. The lack of the runtime adaptation has been highlighted as one of the main drawbacks of IMS LD [16,17].

In this paper, we present a solution for the runtime adaptation of the structure of the IMS LD-based UoLs. Our solution is based on the results presented in [18], where IMS LD models are represented as workflows specified using Petri nets [19,20]. This work approaches each construct of IMS LD (i.e., methods, plays, acts, and activities) with a Petri nets model, and composes these nets by means of the hierarchical Petri nets paradigm. From this model we propose a new mechanism for runtime adaptation based on the substitution of Petri nets. With this contribution, the Petri net, and equivalently the IMS LD learning flow, can be automatically configured to incorporate new plays, acts, and activities in runtime, avoiding to stop the UoL execution to redesign it. The new model has been implemented as part of the OPENET4LD engine [18].

The rest of the paper is organized as follows. Section 2 analyzes recent approaches dealing with dynamic changes during the execution of UoLs. Sections 3 and 4 provide an overview of modeling IMS LD and controlling the execution of the learning flow with Petri net-based workflows. Section 5 details our model for supporting dynamic changes in IMS LD. Section 6 describes the prototype system and Section 7 the results of its application to real UoLs created by educators. Finally, the conclusions and future directions of our work are summarized in Section 8.

2. State of the art of runtime changes in IMS Learning Design

Only few papers deal with this problem in the domain of adaptive learning. One possible cause is the immaturity of this technology, which yet is unable to dominate traditional e-learning systems. Another reason is that some authors consider that adaptive mechanisms provided by EMLs are sufficient to give this support. However, as noted in [21,17], their argumentation is incorrect:

- Defining different ways that take into account all possible variations in the learning flow can be a correct approach when the UoL is small. However, as the complexity increases, the solution is totally unfeasible since the average size of UoLs would grow exponentially. Moreover, the cost of maintenance and modification of one of these UoLs would be important.
- The algebras that capture the semantics of adaptation of the EMLs, such as the algebra of IMS LD *Level B*, are not designed to bear structural changes. Certainly, the specification allows to modify the behavior of the UoL at runtime based on the value of different properties, rules and conditions. However, the change only affects the flow of learning and not the structure of the UoL which remains the same.

A work that improves traditional approaches is proposed by Zarraonandia et al. [21], who present a mechanism to introduce small variations in the original learning designs. Authors extend the algebra given by IMS LD to facilitate the introduction of changes in the conditions, resources, and activities of the UoL in runtime. With this solution authors achieve to reuse the designs without modifying their definition, avoiding the exponential growth of UoLs, but the solution is not designed to support deeper structural changes, such as defining new plays or acts, allowing thus only changes at the level of activity.

None of the main IMS LD players analyzed, CopperCore [8], GRAIL [9], ELeGI [14], Voyager2 [11], COW [12], Tuple-LD [13], MOT+ [10], and LAMS [22], offer the possibility of introducing structural changes during execution. Even workflow-based players (COW and LAMS) do not enable this kind of changes. However, today most workflow management systems support some type of change in runtime. These changes can be managed by a centralized process engine or following a collaborative approach [23], where a set of users dynamically modify the structure of the workflow. For non-collaborative approaches, such as changes carried out by teachers in runtime, the main issue is how to implement the change. Generally, changes can take place at two levels: instance or schema. The first type refers to changes to a single workflow instance, while a schema change may imply to adapt a collection of related instances.

In any case, there is some consensus that the main aspect is to ensure that changes do not cause inconsistencies, such as data loss, or errors, such as deadlocks. In this context, graph-based workflow languages, such as Petri nets, have been most widely used. In [24] Ellis et al. utilize timed flow nets and a technique called *change regions* to determine the correctness of the workflow migration. In [25] Ellis and Keddara improve this approach introducing the concept of flow jumper to move the old workflow to the new workflow. However, authors do not detail how to transfer the data from the old to the new workflow. In [26,27] Agostini and DeMichelis propose a similar approach based on linear jumps and define the criteria for transferring the data, although they restrict the structure that both workflows must have. In [28] van der Aalst improves the change regions calculation identifying both static and dynamic change regions, and introduces the notion of *valid transfer*, which guarantees the soundness in the migration of a (sub)type of Petri nets called workflow-nets. Although more solutions were analyzed (see [29] survey), all presented the same problem: in order to ensure a sound migration, very strong restrictions are required in the structure of the replacement net. This type of approach is very suitable considering that workflows can support a vast variety of control structures and structural changes. However, the IMS LD learning flow model is much more limited in this aspect, and so traditional approaches for assessing workflow changes are too restrictive.

Summarizing, current IMS LD players do not allow instructors to incorporate changes in runtime at any level of granularity of the learning flow (activities, acts, and plays). In turn, although some papers in workflow research have been proposed to modify the structure of Petri nets at runtime, these proposals are not suitable to deal with the issue of changing the structure of IMS LD-based courses in runtime because our structural changes affect Petri nets at a hierarchical level. Taking this into account, in this paper we have defined a hierarchical Petri net model for IMS LD-based UoLs that verifies the necessary properties for making soundness changes in runtime. The large experience of the Petri nets community in the definition of changes in runtime was a main factor for selecting this formalism in our approach. On the contrary, other formal techniques for process modeling, such as process algebra's, do not deal with this problem. The main reason is that runtime changes must take the state of the net into account, both to verify if the change can be performed as to determine the new state after these changes have being performed. However, process algebra's are event-based, i.e., transitions are modeled explicitly and the states between subsequent transitions are only modeled implicitly. Therefore, the modeling of a change model is rather difficult.

3. Petri net model for representing IMS Learning Design

Petri [19] introduced Petri nets in his doctoral thesis as a tool to simulate the dynamic properties of complex systems using graphical