A Model-based Approach to SCORM Sequencing

Mark Melia, Ronan Barrett, and Claus Pahl

School of Computing, Dublin City University, Glasnevin, Dublin 9
firstname.secondname@computing.dcu.ie

Abstract. SCORM has become the de-facto standard in course interoperability, and with the advent of SCORM 2004 course creators can now specify how learning content is sequenced using the SCORM standard. However, for the course creator to understand and use the SCORM sequencing standard he or she must understand the use of and manipulation of SCORM's complicated sequencing rules and conceptual structures. This represents a barrier for most course creators.
In this paper we propose a method for visualising and modelling learning content sequencing using a diagram-based standardised modelling language. Models can then be mapped to a SCORM implementation. By specifying the sequencing within a SCORM package through an intuitive modelling language, it is hoped that the SCORM sequencing functionality will become more accessible.

1 Introduction

Learning content creation is a costly and time consuming task, course creators are increasingly looking to reuse learning content created by others in order to reduce the time and cost involved in establishing and maintaining a course. To aid course creators in sharing and reusing existing courses, standards have emerged which allow for the portability of a course from one Learning Management System (LMS) to another. Of these standards, the Sharable Content Object Reference Model (SCORM) [1] has emerged as the most widely used.

The most recent version of SCORM, SCORM 2004, allows the course creator to include learning content sequencing information. This is an improvement on the previous versions of the SCORM standard as it recognises the importance of learning content sequencing in pedagogy. However, the SCORM sequencing standard is difficult to comprehend and sequencing rules can be very complicated [2] [3]. The complexity involved in applying SCORM sequencing presents the course creator with a barrier when attempting to apply a sequencing strategy to a course. In order for the course creator to overcome this barrier and to take advantage of SCORM sequencing, we present a more intuitive way for the course creator to include sequencing information using a standardised modelling language known as the Unified Modelling Language (UML) [4]. From these models the necessary SCORM constructs are generated. This novel way of manipulating and creating sequencing in SCORM will allow much more non-technical course creators to take advantage of SCORM sequencing.

2 SCORM

SCORM is a collection of specifications and standards that allow for the encapsulation of learning content and pedagogy. The specifications and standards that come together to make SCORM can be grouped under 3 headings. They are, the SCORM Content Aggregation Model (CAM), SCORM Sequencing and Navigation (SN) and the SCORM Run-Time Environment (RTE). From the perspective of sequencing the SCORM CAM and SCORM SN standards are of most interest to us.

The SCORM CAM is a collection of standards used for the packaging and describing of learning resources. An XML file known as the manifest file, which is based on the CAM standard, is used to describe the structure of the learning resources.

The SCORM SN is based on the IMS Simple Sequencing standard [5] and describes how learning content contained within a SCORM package can be sequenced according to learner actions and preferences. The SCORM SN allows the course creator to create a course at design time that behaves in a learner-centric way taking into account such things as prior knowledge and learner preferences at run-time.

3 SCORM Complexity and Limitations

As mentioned in the previous section, SCORM packages have two main parts, the learning resources and the manifest file. The manifest file is based on the IMS Content Packaging XML standard (part of SCORM CAM) [6]. Interwoven into the manifest file is the IMS Simple Sequencing standard [5] (part of SCORM SN), this standard allows the course creator to specify how learning resources in the SCORM package will be sequenced.

IMS simple sequencing is considered a complicated standard by [2] and [3]. The standard is only termed "simple" as it prescribes sequencing for only one user. IMS simple sequencing creates a conceptual activity tree of learning resources, which by default is sequenced using a depth-first tree search algorithm. The activity trees can be divided further into clusters. Clusters are any node in an activity tree and its immediate children. At the cluster level, in an activity tree, general sequencing strategies can be set. These sequencing strategies are known as the "Sequencing Control Mode". The flow of the learner through the activity tree can be altered through a variety of sequencing rules. The course creator also has learning objectives to manage. Each node in the activity tree can be associated with one primary and various secondary learning objectives. The primary objective can also be mapped to a global learning objective. Rules use these objectives in evaluation and set them when fired. Objectives can also be set by events within the SCORM course, such as a learner obtaining a certain mark in a test. In order for a course creator to create a very simplistic sequencing model it is necessary for the course creator to have a competency in all these aspects of simple sequencing. This, we feel, is a lot to ask of course creators, whose primary concern should be in the subject matter and pedagogy and not in conforming their sequencing strategy with a specific XML standard.

Due to the popularity of the SCORM standard and also due to the complexity of the standard a number of tools have been developed which aim to aid the course creator in constructing a course SCORM-compliant.

The Reload editor [7] is one of the most popular SCORM package creation tools. The Reload editor allows the course creator to add learning resources to a new SCORM package with minimal effort. To add a resource to a scorm package the course creator can just drag and drop the resource into the SCORM package in the Reload environment. Resource details can then be added to the manifest file through a dialog box. The Reload editor allows the course creator to apply sequencing techniques using a template system that maps directly to the manifest XML. In order for the course creator to take full advantage of the Reload editor he or she must understand SCORM SN's conceptual framework. Due to SCORM SN's complexity this presents the course creator with a barrier in using Reload in applying SCORM SN sequencing rules and strategies.

4 Modelling SCORM Sequencing

To aid course creators in sequencing course content we propose the use of the UML, a standardised modelling language developed by the Object Modelling Group (OMG). Although the modelling language has been developed for software engineering it can be used to visualise high level sequencing strategies by modelling the SCORM activity tree. The UML can also be used to specify the lower level aspects of sequencing such as the path of the learner through the learning resources given a set of circumstances at runtime.

We propose the use of a standardised modelling language for specifying SCORM sequencing provide us with the following benefits:

- models are intuitive and easily comprehensible, offering a clear abstract visualisation of the sequencing in question
- standardised models allow for the documentation of a sequencing strategy
- allow for different views of a sequencing strategy
- standardised modeling language allow for automated sequencing and packaging of learning resources

Sequencing models act as a compliment to tools such as the Reload Editor. The Reload Editor is very effective in creating the initial SCORM package, but is not very intuitive when the course creator has to specify complicated sequencing strategies. We would envisage our modelling tool as a future plugin for the Reload Editor. Course creators use the reload editor to add learning resources and index the learning resources with in the manifest, and then use the modelling editor in Reload to specify the sequencing strategy.

In order for the modelling language to express the SCORM SN adequately we extend the modelling language to provide us with two types of models. The first one, the content packaging model, allows for a visualisation of the course's

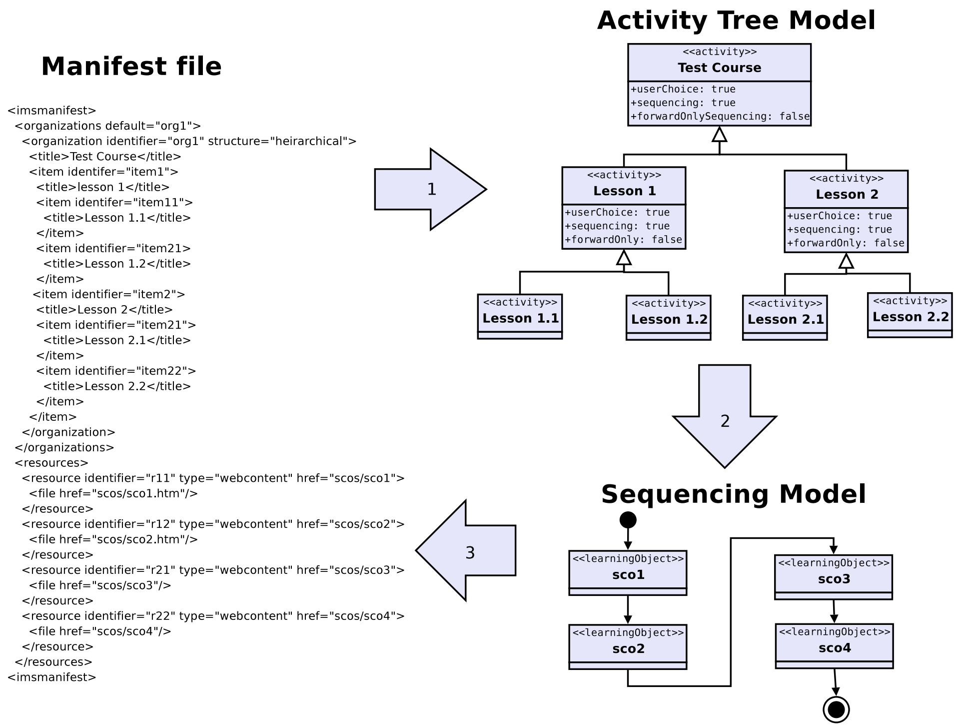

Fig. 1. SCORM Modelling

SCORM activity tree. At this level of abstraction the course creator can set high level sequencing strategies, known as the SCORM sequencing control modes. The second type of model is the sequencing model. This model allows the course creator to visualise how the learner will proceed through the course content. The sequencing model allows the course creator to set more fine-grained sequencing instructions, for example an alternative course sequence should the learner demonstrate a competency in a particular area of the course. The decisions made by the course creator when manipulating the sequencing model have a direct corelation with the SCORM sequencing rules set in the SCORM package's manifest file.

To begin the modelling process the initial SCORM package is built using Reload, or a similar tool. From the manifest the activity tree model is automatically generated as shown in Figure 1 (step 1 in Figure 1). When the course creator is satisfied with the high-level sequencing strategies as set out in the activity tree model, the activity tree model is used to generate the sequencing model (step 2 in Figure 1).

The sequencing model is used to create the fine aspects of sequencing course content. The sequencing model offers the course creator a range of methods to

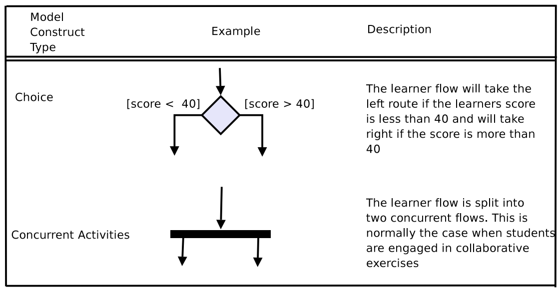

Fig. 2. Learning Activity modelling constructs

specify how the course will be sequenced given a certain circumstance during runtime. Figure 2 highlights two such modelling constructs. When the course creator is happy with the sequencing, the sequencing details are committed back to the manifest file (step 3 in Figure 1).

5 Scenario

In this section of the paper we aim to demonstrate what is involved in setting a range of sequencing requirements for a personalised SCORM-compliant course using the modelling language proposed in this paper. Through an example scenario we hope to highlight the benefits of modelling pedagogical design by demonstrating how some common pedagogical requirements can be mapped to the modelling language.

The course we wish to create is a basic computer programming course consisting of two parts. The first section looks at selection statements, which allow for decision points in computer programming and the second at loops, which allow for particular computer instructions to be executed iteratively. The selection statement section of the course has two more parts, one teaching the if statement and one teaching the switch statement. The learner must be competent in the if statement before proceeding to learn about the switch statement. To ensure this, only learners who score over 50% in an if statement test may proceed to the switch statement. If the learner scores less than 50% they must proceed through the if statement material again.

The section of the course on loops contains two subsequent parts as well, one on the for loop and another on the while loop. It is a requirement of the course that the learner has an overview knowledge of both loops but knows only one in detail. The learner controls what loop they learn in detail.

The learner will have full control of the course and can choose any aspect of the course at any given time. Sequencing flow buttons will be offered to the learner so the learner can follow the lesson as set out by the course creator. Figure 3 demonstrates how this is set using the content packaging model. By setting userChoice to true the learner will be able to select any aspect of the

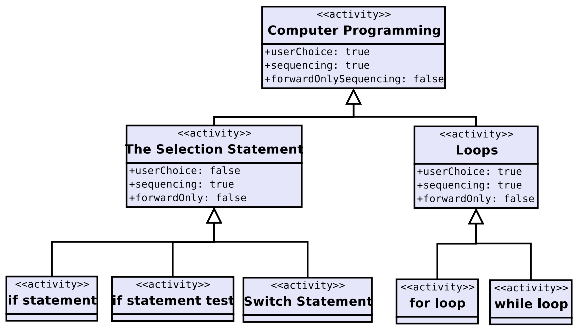

Fig. 3. Modelling high level learner navigation strategy as set out by the course creator

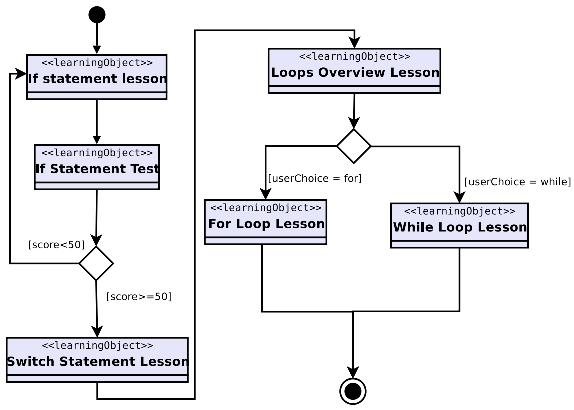

Fig. 4. Modelling sequencing as set out by the course creator

course at his or her discretion. By setting sequencing to true the learner will be provided with a continue navigational button to progress through the course, and by setting forwardOnly to false he or she will be provided with a previous button to work back through aspects of the course already taken.

Figure 4 demonstrates how lower level sequencing is set by the course creator in the sequencing model. It is here the course creator can set the conditions under which the learner may proceed to the switch statement learning content after and only after completing the if statement learning content and successfully completing a test on the if statement. The sequencing model also allows the course creator to express that the learner only requires an overview in loops but must know one type of loop in detail, the type of loop that the learner learns in detail is left to learner preference.

When the course creator is happy the course requirements have been expressed in the models the SCORM package and the manifest file is generated. Figure 5 is a small extract of the manifest file that would be generated for the

```
<item identifier="selection_statements" identifierref="if_res">
 <title>The If Statement</title>
 <item identifier="selection_statements" identifierref="ifass_res">
  <title>If Statement Assessment</title>
 </item>
 <imsss:sequencing>
  <imsss:controlMode choice="true" flow="true" />
   <imsss:sequencingRules>
    <imsss:postConditionRule>
     <imsss:ruleConditions conditionCombination="any">
      <imsss:ruleCondition condition="satisfied" operator="not"/>
     </imsss:ruleConditions>
     <imsss:ruleAction action="Retry All"/>
    </imsss:preConditionRule>
   </imsss:sequencingRules>
   <imsss:objectives>
    <imsss:objective satisfiedByMeasure="true" objectiveID="if_obj">
     <imsss:minNormalizedMeasure>0.5</imsss:minNormalizedMeasure>
    </imsss:objective>
   </imsss:objectives>
   <imsss:deliveryControls completionSetByContent="true" tracked="true"/>
  </imsss:sequencing>
 </item>
 <item identifier="selection_statements" identifierref="switch_res">
  <title>The Switch Statement</title>
 </item>
</item>
<item identifier="sco2" identifierref="sco2_res">
<title>Loops</title>
```

Fig. 5. Scenerio Manifest Extract

computer programming course. It shows how the course requirement that states a learner must achieve a score of more than 50% in the if statement test (or 0.5 in SCORM's objective minimum satisfied normalized measure) before proceeding to the switch statement.

6 Related Work

There is a clear need to make the application of sequencing and navigational strategies in SCORM more intuitive for the course creator. Modelling can satisfy that need by providing the course creator with a simple method to specify course sequencing which can be mapped to a manifest file. Using UML to model SCORM has been investigated in [8], although the author does recognise the potential impact of using UML with SCORM, the modelling methodology is somewhat naive and superficial.

In [9] Jun-Ming Su et. al. use a modelling notation that divides a SCORM course into sequencing objects. Sequencing objects are associated with clusters in the SCORM activity tree. Once sequencing objects have been defined, by the course creator, an engine is used to create the corresponding manifest file. Sequencing objects are excellent examples of how models can be used to alleviate some of the complexity involved in creating a SCORM sequencing. Sequencing objects use a self-defined notation, and are not standardised, and therefore do not reap the advantages of the automated transformation processes developed for standardised modelling languages such as the UML.

7 Conclusions

In this paper we have highlighted how SCORM SN can benefit the course creator, but also how difficult it is for the course creator to realise the potential of the SCORM SN model, highlighting the need for a more intuitive way of specifying SCORM sequencing.

Modelling is a technique used in nearly every sector to represent a complex processes to both people and to machines, due to their intuitive nature. Using a recognised modelling language to model the SCORM SN allows for the course creator to harness the full power of SCORM SN and to create much more complex sequencing structures in a SCORM package.

References

1. Advanced Distributed Learning: SCORM 2004 Overview (2004) Available from http://www.adlnet.gov/scorm/index.cfm.
2. Mackenzie, G.: SCORM Primer: A (mostly) Painless Introduction to SCORM. Technical report (2004) Available from http://www.mcgill.com/media/SCORM_2004_Primer_v1_McGill_Digital_Solutions_Gord_Mackenzie.pdf.
3. Deibler, N.P.: Designing and Developing SCORM Content: Practical tips to get your Project Moving in the Right Direction (2004)
4. Eriksson, H.E., Penker, M., Lyons, B., Fado, D.: UML 2.0 Toolkit. Wiley Publications, Indianapolis, Indiana (2003)
5. IMS Global Learning Consortium: IMS Simple Sequencing Specification (2003) Available from http://www.imsglobal.org/simplesequencing/.
6. IMS Global Learning Consortium: IMS Content Packaging (Version 1.1.3) Overview (2003) Available from: www.imsglobal.org/content/packaging/.
7. The RELOAD Project: (The RELOAD Metadata and Content Packaging Editor) Available from: http://www.reload.ac.uk/editor.html.
8. Hu, S.C.: Application of the UML in modeling SCORM-conformant Contents. In: Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT03), IEEE Computer Society (2005)
9. Su, J.M., Tseng, S.S., Weng, J.F., Chen, K.T., Liu, Y.L., Tsai, Y.T.: An Object Based Authoring Tool for Creating SCORM Compliant Course. In: 19th International Conference on Advanced Information Networking and Applications. Volume 1., IEEE (2005) 209