Web-Based Intelligent Tutoring Systems Using the SCORM 2004 Specification
A Conceptual Framework for Implementing SCORM Compliant Intelligent Web-based Learning Environments

Gustavo Santos, Álvaro Figueira
CRACS & INESC-Porto LA, University of Porto
Department of Computer Science
Porto, Portugal
{gsantos, arf}@dcc.fc.up.pt

Abstract — This paper describes a conceptual framework for implementing Intelligent Tutoring Systems using SCORM 2004. The main objective is to discuss how the SCORM 2004 sequencing and navigation specification can allow the development of Intelligent Web-Based Learning Environments using the sequencing and navigation tracking data, and rule set. Our main argument is that SCORM 2004 sequencing and navigation specification can be used to implement the two main functionalities of an ITS, (1) the inner loop and (2) the outer loop.

Keywords: Intelligent Educational Systems; Learning Systems Platforms and Architectures; Adaptive and Personalized Technology-enhanced Learning;

I. INTRODUCTION

For several years Intelligent Tutoring Systems (ITSs) have been developed and shown to lead to impressive improvement in student learning in a range of domains, including middle school mathematics [1], physics [2] and programming languages [3]. One of the main reasons for the success of these systems is based on their ability to provide adaptive real-time sequencing of learning activities for every student, using a student model that is updated while interacting with the student.

Some of the critics that ITSs have received are related to the fact that their development is very time consuming, and in general they cannot be reused or imported to different platforms. Interoperability and reusability are two of the main benefits of the Sharable Content Object Reference Model (SCORM) [4]. SCORM is a collection of standards and specifications for web-based learning. It allows the development of educational content that is flexible to incorporate instructional components in multiple applications and contexts. SCORM defines communication between client and server, and also defines how content may be packaged into a compressed transferable file.

II. BACKGROUND AND RELATED WORK

A. Intelligent Tutoring Systems

In Figure 1 we present the classic ITS architecture [5], and also the functional model of an ITS. Many authors make reference to this modular architecture where the ITS typically consists of a model of the expert's knowledge, the model of learner's current knowledge, the pedagogical principles of the domain, and the user interface.

As the student proceeds with the material, the student model and the expert model are compared using AI techniques, and the sequence of learning activities is dynamically generated to suit the needs of the student. The modular ITS architecture presented in Figure 1 is subjective and also susceptible to variations. It is actually not necessary to structure code in order to separate the modules and
implement an ITS. When it comes to an ITS, what is most important is the set of functionalities that a tutor must have. Kurt VanLehn published in 2006 [6] a paper describing the basic set of functionalities that every ITS must have. They can be grouped in two major categories, (1) the inner loop, and (2) the outer loop.

The inner loop is basically responsible for giving appropriate feedback and hints when a student is working on an activity. It can also assess the student’s competence and update the student model, which is used by the outer loop to select a next task.

The main duty of the outer loop is to wisely select a task/activity for the student to work. The main design issues are (1) selecting a task intelligently and (2) obtaining a rich set of tasks to select from. Whereas the outer loop is about tasks, the inner loop is about steps within a task. The outer loop executes once per task, while the inner loop executes once per step. Conceptually, an ITS is comprised of two loops as illustrated in the following pseudo-code:

```
Until tutoring is done, do:
1. Tutor poses a task
2. Until the task is achieved, do:
   2.1. Tutor may give a hint
   2.2. Student does a step
   2.3. Tutor gives feedback on the step
3. Student submits the solution to the task
```

A step is a user interface action that is part of completing a task/activity (e.g., solving a problem). Because tutoring systems vary considerably in how they support learning when students are working on steps, let us simplify by assuming that they offer the students services as they work. In particular, the following are some of the most common services:

- Minimal feedback on a step. In most cases, this means indicating only whether the step is correct or incorrect;
- Error-specific feedback on an incorrect step. This information is intended to help the student understand why a particular error is wrong and how to avoid making it again;
- Hints on the next step;
- Assessment of knowledge;
- Review of the solution.

In short, as suggested by the code pseudo-code, we can describe the two main characteristics of an ITS in a nutshell as, (1) to select the next suitable task/activity to the student, and (2) to assist the student to perform the task/activity [6].

B. SCORM 2004 Sequencing and Navigation

SCORM 2004 is composed by the following three specifications: (1) Content Aggregation Model (CAM), which specifies how content should be packaged and described; (2) Run-Time Environment (RTE), which specifies how content should be launched and communicate with the LMS; and (3) Sequencing and Navigation (SN), which specifies how the learner can navigate between parts of the course (SCOs). In this paper we only give details about the SN specification.

In SCORM, sequencing is what happens between SCOs and outside of SCOs [7]. When a learner exits a SCO, sequencing is responsible for determining what happens next. SN orchestrates the flow and status of the course as a whole. In SN, aggregations and SCOs are referred by the generic term “activity”. Everything in sequencing is an activity. Activities are nested in parent-child relationships that form them into an “activity tree”.

Every activity has two sets of data associated with it, (1) tracking data and (2) a sequencing definition. The tracking data represents the current state of the activity (its status, score, etc.). The sequencing definition is a set of rules that define how the activity should be sequenced.

Conceptually the activity tracking model can capture in run-time data corresponding to completion status, satisfaction status, progress measure and score (like an ITS inner loop). It also tracks some data specifically related to the innards of the sequencer, such as the number of attempts on the activity.

III. SCORM COMPLIANT ITSs

A. Sequencing Process

The actual process of sequencing occurs whenever a course is launched, whenever a SCO exits, or whenever the learner makes a navigation request through the LMS’s user interface. When the sequencer is invoked the LMS then starts the “sequencing loop”.

The sequencing loop is a set of defined algorithms that apply the sequencing rules to the current set of tracking data to determine which activity should be delivered next. The sequencing loop may result in an activity to be delivered, it may result in the course exiting or it may result in an error condition. Conceptually, the algorithm for the sequencing loop is:

```
End the current activity {
   transfer the run-time data to the activity-tree;
   roll up state data to all clusters in the activity tree;
}
Check from sequencing rules what activities could be delivered;
Determine which activity should be delivered next;
Deliver the identified activity to the learner;
```

The previous work of Kazi [8] about SCORM compliant Intelligent Tutoring Systems makes the following analogy between ITS modules and SCORM modules:

“If we analyze the functional model of an ITS system, all of its modules can easily be substituted by SCORM-2004 specification”...“Communication module falls under the SCORM RTE”...“SCORM SN together with SCORM CAM can handle the functionalities of the pedagogical module”...“Student model can be implemented through Tracking Status Model and SCORM RTE Data Model”...“Since the Expert module is very much subjective, this should be embedded with leaf activities or actual tutoring problems... plugged-in as an Intelligent Tutoring Applet”...

Kazi’s solution is plausible. However, a few things should be discussed. He suggests including in the leaf activities Web applications (like applets, or flash files) to handle the expert model locally and provide the activities intelligent tutoring capabilities. This would solve part of the problem (the inner loop), but the details of how to achieve it are not described. SCORM 2004 SN has means for
implementing the inner loop using the SCORM API and SCORM objectives to record data about the user.

Kazi also mentions in his approach that SCORM SN module can handle together with the SCORM CAM module the function of the ITS pedagogical module. One of the functions of the pedagogical module is performing the outer loop. However, Kazi also does not describe in detail how this could be performed. In our approach we show how the outer loop can be achieved using the SCORM 2004 tracking data and sequencing rule set.

B. Sequencing and Navigation and ITSs

Our approach for SCORM compliant Web-based ITSs relies on using the SCORM 2004 SN specification. We do not ground our approach on mapping ITS modules to SCORM modules, but on using SCORM 2004 SN for implementing what defines an ITS in terms of functionalities [6], outer loops and inner loops.

First of all, SCORM 2004 SN specification defines that every activity has two sets of data associated with it, (1) tracking data and (2) sequencing definition [8]. The tracking data can be used for registering what a student knows about the domain (user model). We can acquire this data while a student works on an activity using the SCORM RTE functions for working with SCORM objectives. Therefore, we have a foundation for implementing inner loop tasks such as updating the student model.

The sequencing definition is a set of rules that define how an activity should be sequenced. SCORM objectives could, and should be used by the rules to influence sequencing. As a result, the tracking data in conjunction with the sequencing rules would have the necessary mechanism to perform task selection. And so, we can execute an ITS outer loop.

With SCORM 2004 SN we can build and deliver activities able to execute in run-time inner loops with intelligent feedback defined by instructional designers. With SCORM objectives the inner loops can store data about the student competence. The SCORM sequencer can use this data to select the next task, implementing therefore outer loops.

Being able in run-time to acquire and store information about what the student has been doing while working on an activity, in conjunction with the possibility to use this information for selecting the next activity are the two basic conditions for being able to implement an ITS. So far we have built already a small prototype of a middle school math ITS this way.

For example, a student is practicing to solve math problems in a SCORM course. He first logs in and an initial math problem is loaded to the user interface. While he is solving the problem, the tracking model is registering data about the activity. The tracking data can therefore be used to provide problem solving assistance to the student (such as feedback indicating whether a step is correct or incorrect, feedback on an incorrect step, a hint, etc.). Once the student finishes solving the problem and leaves the SCO, the sequencing mechanism will use the tracking data and the sequencing rule set to choose the next suitable problem/activity for this student (steps 1 to 7 in Figure 2).

IV. CONCLUSIONS

In this paper we have presented a conceptual framework for implementing SCORM compliant Web-based ITSs. We do not ground our approach on mapping ITS modules to SCORM modules, but on using SCORM 2004 SN for implementing what defines an ITS in terms of functionalities [6], outer loops and inner loops.

Both loops can be implemented with the SCORM 2004 SN tracking data and sequencing rule set using the SCORM RTE functions for working with SCORM objectives. As a result, one with expertise in building ITSs can develop online courses presenting these two essential loops/functionalities of an ITS. As a reminder, SCORM 2004 does not solve issues of building ITSs. To build SCORM compliant ITSs we still have to face the same issues of building a usual ITS, such as knowledge engineering, user modeling, and defining pedagogical strategies for teaching a particular subject.

REFERENCES


