Seqins – A Sequencing Tool for Educational Resources

Ricardo Queirós[1], José Paulo Leal[2], and José Campos[3]

1 **CRACS & INESC-Porto LA & DI-ESEIG/IPP, Porto, Portugal**
ricardo.queiros@eu.ipp.pt
2 **CRACS & INESC-Porto LA, Faculty of Sciences, University of Porto, Portugal**
zp@dcc.fc.up.pt
3 **Lusíada University**
jjscampos@eu.ipp.pt

Abstract

The teaching-learning process is increasingly focused on the combination of the paradigms "learning by viewing" and "learning by doing." In this context, educational resources, either expository or evaluative, play a pivotal role. Both types of resources are interdependent and their sequencing would create a richer educational experience to the end user. However, there is a lack of tools that support sequencing essentially due to the fact that existing specifications are complex. The Seqins is a sequencing tool of digital resources that has a fairly simple sequencing model. The tool communicates through the IMS LTI specification with a plethora of e-learning systems such as learning management systems, repositories, authoring and evaluation systems. In order to validate Seqins we integrate it in an e-learning Ensemble framework instance for the computer programming learning.

1998 ACM Subject Classification K.3 Computers and Education; K.3.1 Computer Uses in Education; Computer-managed instruction (CMI)

Keywords and phrases e-Learning, Learning management systems, Automatic evaluation

Digital Object Identifier 10.4230/OASIcs.SLATE.2013.83

1 Introduction

In the last few years, higher education institutions have been challenged by the emergence of new information and communication technologies (ICT). In this context, e-learning has been adopted as the preferred channel for the dissemination of knowledge.

In the mid-1980s, educational researchers found that learning was faster and better when some practice/evaluative resources were replaced by worked-out examples that demonstrated the lesson skills. Sweller and Cooper [10] found that errors were reduced in half when 12 math practice resources were replaced by 6 worked-out examples, each followed by practice.

Many experiments showed a combination of examples and practice exercises engages novice students rather than just practice. For instance, in a lesson on how to use programming variables, an example showing the steps to declare and initialise a variable would be followed by a practice in which the learner must repeat these steps in a slightly different situation. When viewing an example, the learner's working memory is free to build a new mental model of the skill. Then the learner can try out the new mental model in a practice problem [3].

Based on these thoughts, Sweller and Cooper [10] used a learner-centered approach to define a constructivist learning model, depicted in Figure 1, based on two learning paradigms: "learning by viewing" and "learning by doing." In this model educational resources, either expository or evaluative, play a pivotal role.

2nd Symposium on Languages, Applications and Technologies (SLATE'13).
Editors: José Paulo Leal, Ricardo Rocha, Alberto Simões; pp. 83–96

OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

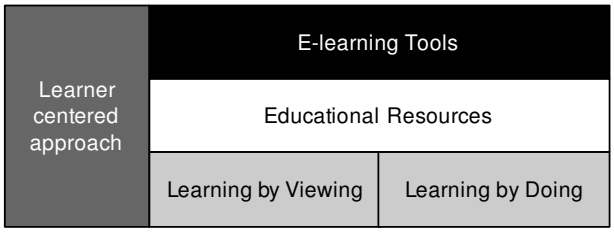

Figure 1 Sweller and Cooper constructivist learning model.

These resources are crucial for the teaching-learning process. It is important that an e-learning system provides a collection of resources covering a course syllabus and with different levels of difficulty. It has been shown that this can improve the performance of students and their satisfaction levels [12]. Students with lower computer skills can begin by viewing examples and solving easier problems in order to progressively learn and to stay motivated to read and solve harder topics later [6]. At the same time this gives them experience which is one of the factors that has a greater influence on student success in learning [13]. In recent years, a large number of educational resources have been developed and published and mostly of them stored in proprietary systems for their own use. Although some standard organisations (e.g. ISO/IEC, IMS, ADL) have created specifications (e.g. IEEE LOM, IMS CP, SCORM, IMS CC) for the description and aggregation of educational resources, each one has its own format, making it difficult to share among instructors and students. This poses several issues on the resources interoperability among e-learning systems.

At the same time, both types of resources are interdependent and their sequencing would create a richer educational experience to the end user. However, there is a lack of tools that support sequencing essentially due to the fact that existing specifications are complex.

In this paper we present Seqins as a sequencing tool for educational resources. Seqins is based on a simple sequencing model where students after seeing a set of examples on a particular topic, consolidate their knowledge solving a set of exercises on the same topic. The student must solve an (a set of) exercise(s) before proceeding to next topic. This approach has several advantages. The most important is the ability to foster heterogeneous classes, enabling students with different learning rhythms to progress autonomously.

The remainder of this paper is organised as follows. Section 2 enumerates the standards and specifications for sequencing resources and integrating tools in e-learning environments. In the following section we present Seqins with emphasis on its sequence, data and communication model. Then, to evaluate the sequencing tool, we present its integration on a network for the computer programming learning. Finally, we conclude with a summary of the main contributions of this work and a perspective of future research.

2 E-learning Specifications

2.1 E-learning Sequencing Specifications

Sharable Content Object Reference Model (SCORM) is a collection of standards and specifications for web-based e-learning. It allows the communication between client side content and a host system called the run-time environment, which is commonly supported by a learning management system. SCORM also defines how content may be packaged into a transferable ZIP file called “Package Interchange Format.” Despite its enormous popularity

Listing 1 IMS SS excerpt within a IMS CP manifest.

```
<manifest ...>
 <metadata>...</metadata>
 <organizations default="TOC1">
  <organization identifier="TOC1">
   <title>Photoshop Tutorial</title>
    <item identifier="ITEM45" identifierref="QUESTION6">
     ...
     <imsss:sequencing>
      <imsss:controlMode choice="false" forwardOnly="true"/>
      <imsss:rollupRules>
       <imsss:rollupRule childActivitySet="all">
        <imsss:rollupConditions>
         <imsss:rollupCondition condition="attempted"/>
        </imsss:rollupConditions>
        <imsss:rollupAction action="completed"/>
       </imsss:rollupRule>
      </imsss:rollupRules>
     </imsss:sequencing>
    </item>
    ...
   </organization>
  </organizations>
  <resources>
   <resource>...</resource>
   ...
  </resources>
</manifest>
```

some researchers have questioned the instructional value of the SCORM model. The focus of the critics was whether atomic de-contextualized learning objects (LO) can support an effective pedagogic goal and whether it could convey an unified learning experience. In order to address these concerns SCORM included the IMS Simple Sequencing specification (IMS SS).

The IMS SS is a specification used to describe paths through a collection of learning activities. The specification declares the order in which learning activities are to be presented to the learner and the conditions under which a resource is delivered during an e-learning instruction. The IMS SS organizes and sequences the package items in a XML manifest section called Organizations. This section aims to design pedagogical activities and to articulate the sequencing of instructions. By default, it uses a tree-based organization of learning items pointing to the resources (assets) included in the package.

Unfortunately, IMS Simple Sequencing does not make the instructional designer's task of bringing instructional order and meaning through sequencing easy. The standard is highly technical and procedural, and lacks support for associating instructional meaning with the sequencing code (e.g. domain related segments). In fact, the IMS SS specification has been the target of much criticism such as:

- Modularity - included in the Learning Objects (IMS CP);
- Scope - widening the scope exaggerated ("spray and pray");
- Utility - More focused on quantity than quality ("shovelware");
- Complexity - difficult to implement.

The IMS CP manifest excerpt in listing 1 shows some of the IMS SS tags to define the conditions under which a specific item is delivered during an e-learning instruction.

IMS attempts to address this pedagogical issue by supplementing its simple sequencing standard with the IMS Learning Design standard. The IMS LD specification is a meta-language for describing pedagogical models and educational goals that can be constructed (e.g. Reload, LAMS) and rendered (e.g. CopperCore, .LRN). However (once again) this standard provides templates for learning in a multi-person/entity setting and is not really applicable to the web-based, single learner setting which is the focus of SCORM 2004.

2.2 E-learning Integration Specifications

Data integration is the simplest and most popular form of integration in content management. This type of integration uses the import/export features of both systems and relies on the support of common formats.

The major LMS vendors include also APIs to allow developers to extend their predefined features through the creation of plug-ins. Blackboard uses the Building Blocks technology to cover the integration issues with other systems allowing third parties to develop modules using the Building Blocks API. The new Moodle versions (from v.2.0 released in November 2010) includes several API to enable the development of plug-ins by third parties such as the Repository API for browsing and retrieving files from external repositories; and the Portfolio API for exporting Moodle content to external repositories. These two API are based on the File API - a set of core interfaces to allow Moodle to manage access control, store and retrieve files. The new File API aims to enhance file handling by avoiding redundant storage. A common interoperability standard that is increasingly supported by major LMS vendors is the IMS LTI specification. The IMS LTI provides a uniform standards-based extension point, allowing remote tools and content to be integrated into LMSs. The main goal of LTI is to standardize the process of building links between learning tools and the LMS. There are several benefits from using this approach: educational institutions, LMS vendors and tool providers by adhering to a clearly defined interface between the LMS and the tool, will decrease costs, increase options for students and instructors when selecting learning applications and also potentiate the use of software as a service (SaaS). The LTI has 3 key concepts as shown in Figure 2 [4]: the Tool Provider, the Tool Consumer and the Tool Profile.

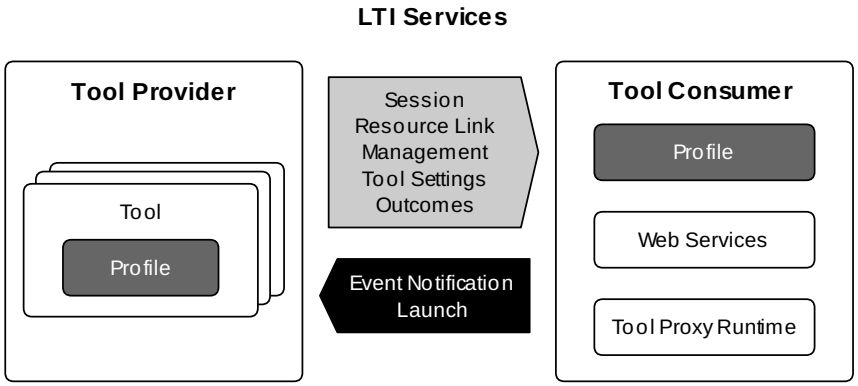

Figure 2 IMS Full LTI.

The tool provider is a learning application that runs in a container separate from the LMS. It publishes one or more tools through tool profiles. A tool profile is an XML document describing how a tool integrates with a tool consumer. It contains tool metadata, vendor information, resource and event handlers and menu links. The tool consumer publishes a Tool Consumer Profile (XML descriptor of the Tool Consumer's supported LTI functionality that is read by the Tool Provider during deployment), provides a Tool Proxy Runtime and exposes the LTI services.

A subset of the full LTI v1.0 specification called IMS Basic LTI exposes a single (but limited) connection between the LMS and the tool provider. In particular, there is no provision for accessing run-time services in the LMS and only one security policy (OAuth protocol[1]) is supported. For instance, to export content from Moodle to Mahara using the Basic LTI the teacher (or LMS administrator) must first configure the tool (Mahara) as a Basic LTI tool in the course structure. When a student selects this tool, Moodle launches a Mahara session for the student. The web interface for this session can either be embedded in Moodle's web interface as an iframe or launched in a new browser window.

Recently, IMS launched the Learning Tools Interoperability v1.1.1 (released in July 2012) that combines Basic LTI and LTI into just Learning Tools Interoperability. This version includes updates and clarifications as well as support for an outcomes service and bidirectional communication support. In this version, LTI has also support for the TP to call IMS Learning Information Services (LIS) when those services can be made available to the TP. LTI does not require LIS services, but the TC can send LIS key information to the TP using values in the basic launch request.

Comparing these three approaches [1] one could say that data integration is the best option when the development effort must be kept to a minimum or no one with technical skills (specially programming skills) is available, since the other two strategies require them. This strategy has also the advantage of not coupling the two systems and enabling a bi-directional communication.

API integration is best suited when batch integration is required since the other two strategies involve the use of the GUI of both systems. For instance, if the work of the students of a given set of courses must be copied on a regular basis from the LMS to their portfolios then the API strategies are recommended. The major drawbacks of this approach are the amount of development required and the tight coupling between the LMS and the other e-learning system, since special plug-ins must be implemented and API are vendor specific.

Tool integration is arguably the best choice in general since it provides a good balance between implementation effort and coupling and security. This is especially true if only unidirectional communication is required and Basic LTI is used. This tool integration flavour is simple to implement and is already supported by most LMS vendors. If bidirectional communication is required then full LTI is needed but in this case the implementation is harder and few LMS vendors support this flavor of the specification. In both cases, tool integration has the added value of providing some basic security features based on the OAuth protocol aiming to secure the message interactions between the Tool Consumer and the Tool Provider.

[1] OAuth security protocol: `http://oauth.net/`

3 Seqins

This section presents Seqins as a sequencing tool for educational resources categorized as:

- Expositives (e.g.: videos, PDF, HTML + images);
- Evaluatives (e.g.: assignments, tests, exercises).

Seqins follows a simple sequencing model. After seeing examples on a particular topic, students consolidate their knowledge by solving a set of exercises related to topic. The success in solving exercises leads the student to the next topic.

The integration of Seqins with other e-learning systems is straightforward since Seqins supports the IMS LTI 1.1. specification who has been increasingly supported by e-learning systems, especially by LMSs.

The following subsections present Seqins architecture and main components, and describe its sequence, data and communication models.

3.1 Architecture

Seqins is a web application for sequencing educational resources that mediates between an LMS and an Evaluator. The role of the LMS is to manage instruction and it will initiate a session in Seqins to sequence a course for a particular student. During the interaction with the student Seqins presents content (HTML, PDF, videos) embed in its web interface. The evaluation of exercises requires the use of a specialised e-learning system designated here as Evaluator. The communication of Seqins with these other systems relies on the LTI protocol described in the precious section.

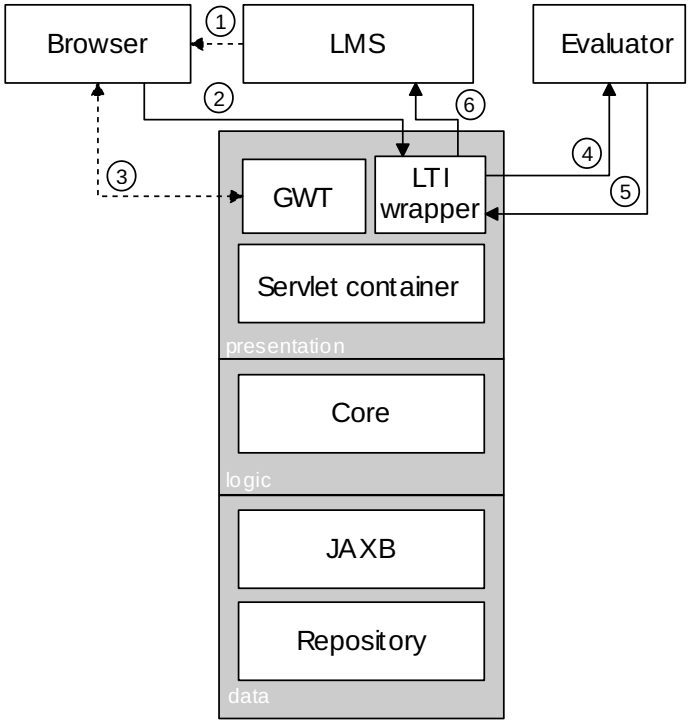

Figure 3 Seqins architecture.

Figure 3 presents the architecture of Seqins following a 3-tier model, divided in presentation, logic and data, highlighting the interaction with the surrounding systems, namely the LMS, the web browser and the automatic evaluator. In this diagram the LTI interactions are represented by solid arrow while plain HTTP interaction are represented by dotted arrows. For better understanding these arrows are marked with a number within a circle representing the sequence of invocations.

A typical use of Seqins starts with a HTTP message replied by the LMS to the student's browser (1) that starts an LTI request processed by the LTI wrapper (2). This request start a Seqins course for the student and creates a GWT interface (more on GWT later on) on the browser where the students interacts with the system. During this interaction, starting an exercise triggers an LTI request to the evaluator (4) handled by LTI Wrapper. Eventually the Evaluator answers with an LTI communication carrying a grade (5) and this information is processed by the core and persistently stored. The steps (4) and (5) are repeated for each attempted exercise. Finally, the results obtained by the student in the course are reported to the LMS using LTI.

One of the key components in this architecture is the LTI Wrapper that implements both sides of the LTI communication. This component receives LTI requests from both the LMS and the Evaluator, and also invokes their services. This Java package was developed for Seqins but can be used by any Java application requiring LTI communication. Both the LTI Wrapper and the GWT user interface are supported by a java servlet container.

The GUI component was developed using an Ajax framework called Google Web Toolkit (GWT). GWT is an open source Java software development framework that allows a rapid development of AJAX applications in Java. When the application is deployed, the GWT cross-compiler translates Java classes of the GUI to JavaScript files and guarantees cross-browser portability. The controls required by GUI are provided by SmartGWT, a GWT API's for SmartClient, a Rich Internet Application (RIA) system. The GWT code is organised in two main packages: the back-end (server) and the front-end (client). The back-end includes all the service implementations triggered by the user interface. These implementations are centralised in the Core component that relies on the LTI wrapper which implements the LTI 1.1 specification.

The data layer deals with the data persistence through JAXB (Java Architecture for XML Binding). JAXB allows Java developers to map Java classes to XML representations providing two main features: the ability to marshal Java objects into XML and the inverse, i.e. to unmarshal XML back into Java objects. In the following subsections we detail these XML representations.

3.2 Data Model

Seqins organizes its data based on two entities: student and course. Since Seqins follows a learner-centered approach it is important to store the learner data. This data is gather by the Tool Producer (Seqins) upon a launch request from the Tool Consumer (typically an LMS). The LTI variables sent in the request are organized in the following data levels: Course, Resource, User, Context and OAuth. In this case two levels are used: the data level is used to store/update information (e.g. identification, name, email) about the user; and the resource level to assign to the respective learner the last resource viewed/solved. Most of this information is stored in the filesystem in XML documents one for each user. Listing 2 shows one of these files.

Listing 2 XML representation of a learner.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<student id="2">
   <href>http://ensemble.dcc.fc.up.pt/crimsonHex/lo/unit01/2</href>
   <lastRequest>2012-12-10T17:18:27.464Z</lastRequest>
   <lastUpdate>2012-11-27T17:22:15.335Z</lastUpdate>
</student>
```

The root element student has an id attribute that identifies the student. The element contains three sub-elements:

- *href* - is the URL of the last resource viewed/solved;
- *lastrequest* - is a timestamp of the last request to see if there has been any change in student status;
- *lastupdate* - corresponds to the last time that a change occurred. In order to not generate too many requests we check for any update after the last request.

The course entity formalizes how a course is structured. The formalization of the course controls the generation of the tree that appears in Seqins's GUI to enable navigation through the course resources. The formalization of a course is made by the definition of a XML Schema depicted in Figure 4, and Listing 3 shows a valid XML instance of a course.

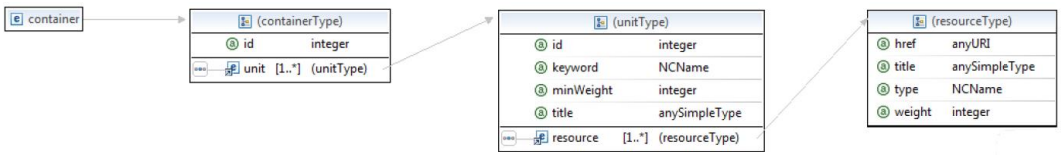

Figure 4 XML representation of a course.

A *course* element contains one or more *unit* elements. Each unit contains a set of *resource* elements. Each resource has four attributes: *title* – name of the resource; *type* – type of the resource (*exp* – expositive and *eval* – evaluative); *weight* – weight for evaluation purposes; and *href* - URL pointing to the resource on the Web.

This XML representation contains two important attributes: weight (w) and minWeight (mw). These attributes operate at resource and unit level respectively and are used by Seqins to determine the progress of a learner within a course. The sequencing model is quite simple:

1. Weights are assigned to the evaluative resources (R) of an unit;
2. Within a unit the visualization/solving is sequential;
3. In order to obtain success in an unit U' the sum of all weights of the successfully solved resources must be greater than or equal to the minimum weight of the associated unit.

$$\sum_{i=1}^{U'length} wR_i \in U' \geq mwU' \tag{1}$$

This model fosters competitiveness since the success in a unit unlocks the access to the following, as if it were a computer game. This feature is assumed by the authors and it has its own risks. Competitive learning is a learning paradigm that relies on the competitiveness of students to increase their programming skills [2, 9], although the concept of "winners and losers" may hinder the motivation of some students [11].

Listing 3 XML representation of a course.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<course title="C# Programming" author="Jose Campos" date="15-03-2013">
 <unit id="1" title="Intro C#" minWeight="70">
  <resource
    title="First program - Hello World!"
    type="exp" weight="0"
    href="http://ensemble.dcc.fc.up.pt/crimsonHex/lo/unit01/1" />
  <resource
    title="Hello ESEIG!!"
    type="eval" weight="40"
    href="http://ensemble.dcc.fc.up.pt/petcha/unit01/1" />
  <resource
    title="Constants and arithmetic operators"
    type="exp" weight="0"
    href="http://ensemble.dcc.fc.up.pt/crimsonHex/lo/unit01/2" />
  <resource
    title="Area of a circle"
    type="eval" weight="30"
    href="http://ensemble.dcc.fc.up.pt/petcha/unit01/2" />
  </unit>
  ...
</course>
</student>
```

3.3 Communication Model

The integration of Seqins with other e-learning systems relies on the LTI specification. The LTI specification recommends REST as the web service flavour for exchanging data among e-learning tools. The LTI functions are summarised in Table 1.

Table 1 LTI functions.

Function	REST	LTI	
		Basic	Full
`Launch`	`POST TA_URL < LTI_PARAMETERS`	yes	yes
`ReplaceResult`	`POST LIS_OUTCOMES_URL < LIS_SOURCE_ID + GRADE`	no	yes
`ReadResult`	`POST LIS_OUTCOMES_URL < LIS_SOURCE_ID > GRADE`	no	yes
`DeleteResult`	`POST LIS_OUTCOMES_URL < LIS_SOURCE_ID`	no	yes

The `Launch` function allows the execution of a particular external tool within the LMS. Two steps are required before launching Seqins from the LMS: 1) the teacher (or LMS administrator) must configure Seqins as an external tool in the LMS control panel by setting the name and the URL of the external tool; 2) the teacher must add an activity to the course structure referring the external tool. Later on, when a student selects the external tool, the LMS uses the URL to launch Seqins through an HTTP POST. This request includes a set of launch parameters (`LTI_PARAMETERS`) as hidden form fields. Table 2 organizes the most important parameters in four groups.

Table 2 LTI launch parameters.

Groups	Variables	Description
Resource	`resource_link_id`	Unique identifier of a resource
	`resource_link_title`	A title for the resource
	`resource_link_description`	A description for the resource
User	`user_id`	Unique identifier of a user
	`user_image`	URI for an image of the user
Context	`context_id`	Context id of the link being launched
	`context_title`	A title of the context
	`context_label`	A label for the context
LIS	`lis_person_name_full`	Full name of the user
	`lis_person_contact_email_primary`	E-mail of the user
	`lis_outcome_service_url`	Unique identifier of the launch
	`lis_result_sourceid`	Outcomes service URL of the TC

This list can be extended by adding custom parameters. The syntax is

`custom_keyname = value`.

Three new parameters were added to the launch request: `custom_collection_id` — defines a link for a collection of exercises in an IMS DRI repository; `custom_sequencing` — defines if the exercises should be solved sequentially; `custom_time_limit` - defines a date/time limit for the solving of all exercises. Listing 4 shows a subset of the launch parameters that the LMS (Tool Consumer) sends to Seqins (Tool Provider).

In this example, Seqins presents the exercises of the `vectors` collection and students should solve them sequentially till November 7, 2011.

Table 1 also refers to three functions included in the IMS LIS Outcomes Service. These functions use the `lis_result_sourceid` parameter included in the launch request that is unique for every combination of `resource_link_id` / `user_id` parameters and identifies a unique row and column within the TC gradebook. After computing a grade, the TA calls the LTI Basic Outcomes Service using the URL stated in the `lis_outcome_service_url` launch parameter. The service supports setting, retrieving and deleting of LIS results associated with a particular user/resource combination (`lis_result_sourceid` parameter). The `replaceResultRequest` function sets a numeric grade (0.0 - 1.0) for a particular result. The `readResultRequest` function returns the current grade for a particular result. The `deleteResultRequest` function deletes the grade for a particular result.

Listing 4 LTI launch with default parameters.

```
1 resource_link_title = Sum two vectors
2 lis_person_name_full= Pimenta Ana
3 roles = Student
4 context_title = Algorithms and Programming
5 lis_result_sourceid={"data":{"instanceid":"1","userid":"2","launchid":1914382991},"hash":"..."}
6 lis_outcome_service_url=http://crimsonhex.dcc.fc.up.pt:8080/moodle/mod/lti/service.php
7 custom_collection_id = http://crimsonhex.dcc.fc.up.pt:8080/crimsonHex/lo/myCollection/vectors
8 custom_sequencing = true
9 custom_time_limit = 2011-11-07 12:00:00
```

4 Integration on Ensemble Framework

For validation purposes, Seqins was integrated with an instance for computer programming of an e-learning framework called Ensemble [7].

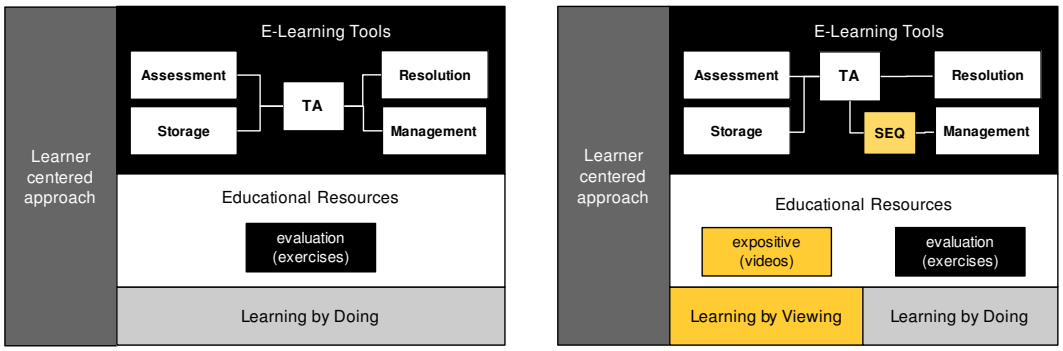

Figure 5 Evolution of Ensemble framework instance to the Sweller and Cooper model.

Ensemble is a conceptual tool to organize networks of e-learning systems and services based on international content and communication standards. Ensemble is focused exclusively on the teaching-learning process. In this context, the focus is on coordination of educational services that are typical in the daily lives of teachers and students in schools, such as the creation, resolution, delivery and evaluation of assignments.

The Ensemble instance for the computer programming domain relies on the practice of programming exercises (evaluative resources) to improve programming skills. This instance includes a set of components for the creation, storage, visualisation and evaluation of programming exercises orchestrated by a central component (teaching assistant) that mediates the communication among all components. The integration of Seqins in this framework instance complies with the model advocated by Sweller and Cooper (Figure 5).

This integration is made at two levels: **communication** and **data**. At communication level Seqins implements the LTI 1.1 specification for the communication with other e-learning systems. The components diagram of Seqins is depicted in Figure 6.

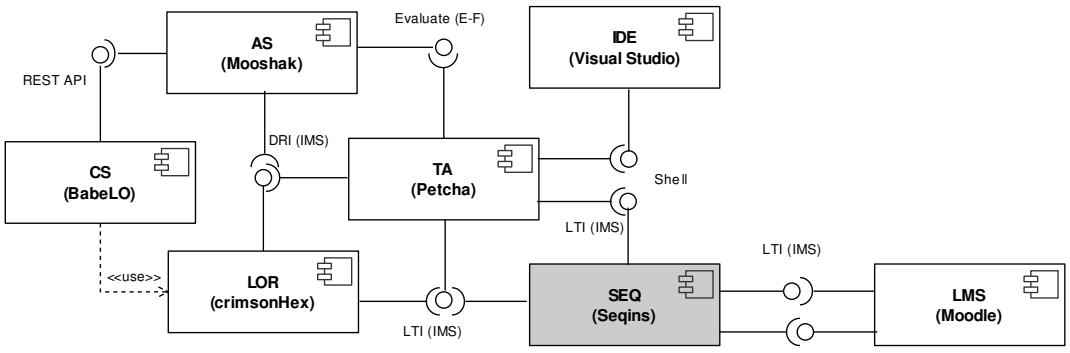

Figure 6 Seqins components diagram.

The integration of Seqins with the LMS and the Teaching Assistant (TA) relies on the LTI specification. After the launch of Seqins through the LMS the LTI parameters are sent by HTTP POST to Seqins. Then, if the learner select an evaluative resource, the same parameters are sent by HTTP POST to the TA. The TA has two main tasks in this setup: to assist teachers in the authoring exercises and to help students in solving them. When a student submits an exercise for evaluation, the grade is sent back to Seqins. This grade will influence the progress of the student. At same time Seqins clones the HTTP response of the TA and sends it back to LMS in order to feed the LMS's grade book.

At the data level, Seqins uses two types of resources: expositives and evaluatives. These resources can be seen in Figure 7 in the left panel.

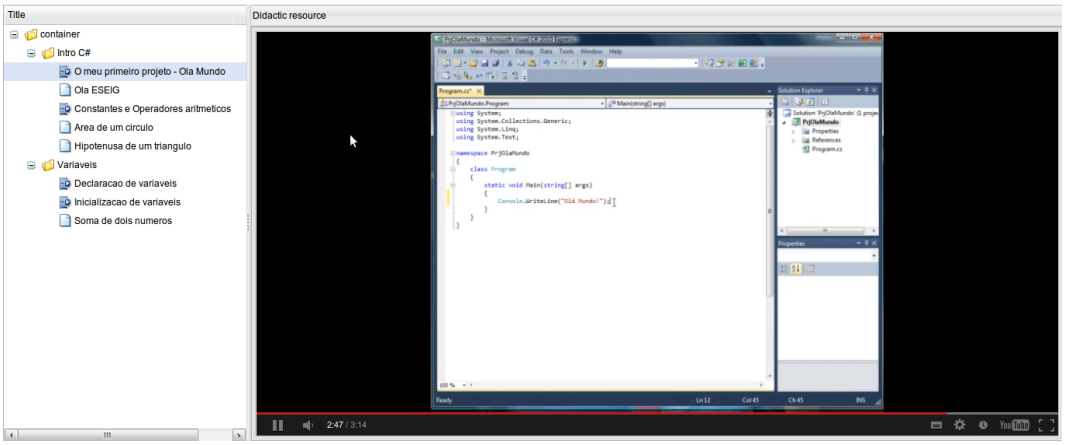

Figure 7 Seqins screenshot.

The former are typically videos (Figure 8) with working examples of exercises solving. The videos were created with Camtasia, a software that records screen activity and voice.

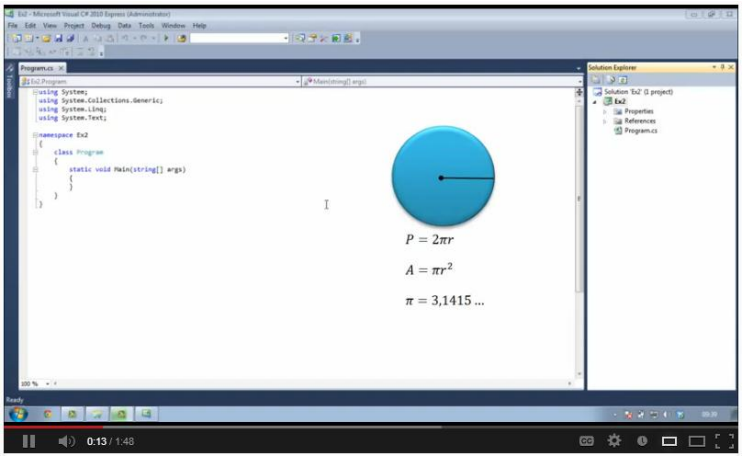

Figure 8 Video screenshot.

The videos were deployed on Youtube. The design requirements for the videos were:

- Coverage - covering all the curricula;
- Diversity - several difficulty levels;
- Fragmented - "video clip time" ($\leq$ 5 minutes);
- Complete - composed by pictures, sound, subtitles, layers.

The evaluative exercises comply with the IMS CC package specification and were extended through an interoperability language called PExIL [8]. PExIL describes the programming exercise life cycle from its creation to its evaluation. The resources were created in the Teaching Assistant and stored in a IMS DRI compliant repository (e.g. CrimsonHex [5]).

5 Conclusions and Future Work

Seqins is a sequencing tool for educational resources. It has a simple sequencing model that depends on the score obtained by the learner on solving exercises. Seqins can communicate with any tool that supports the IMS LTI specification. In order to evaluate this feature, Seqins was integrated with an Ensemble instance for computer programming learning. In this context, learners can launch Seqins through the LMS and browse resources of two types: expositive and evaluative. The former are typically short videos with workout examples on solving programming exercises. The latter are programming exercises that learners should solve on their favourite IDE guided by the pivot component of Ensemble, the Teaching Assistant.

The main contributions are the design and implementation of a sequencing tool based on a simple sequencing model. The LTI integration is also detailed and can prove to be useful for others educators with a similar sequencing goal in heterogeneous environments.

As future work we intend to:

- Create more expositive resources (e.g. C# programming course and other courses);
- Improve sequencing model (e.g. optative - "solve N of X" and conditional - "If Then Else");
- Improve graphical user interface (e.g. sophisticated graphical controls through SmartGWT);
- Include a competitive facet (e.g. statistics on number of people that solved a particular exercise).

References

1 *Integration of ePortfolios in Learning Management Systems*. Springer Verlag, 2011.

2 Juan C. Burguillo. Using game theory and competition-based learning to stimulate student motivation and performance. *Comput. Educ.*, 55(2):566–575, September 2010.

3 Ruth Colvin and Clark Richard. Learning by viewing versus learning by doing : Evidence-based guidelines for principled learning environments. *Performance Improvement*, 47(9):5–13, 2008.

4 T. Gilbert. Leveraging sakai and ims lti to standardize integrations. In *10th Sakai Conference*, 2010.

5 José Paulo Leal and Ricardo Queirós. Crimsonhex: a service oriented repository of specialised learning objects. In *ICEIS 09 - 11th International Conference on Enterprise Information Systems, Milan, Italy*, volume 24 of *Lecture Notes in Business Information Processing*, pages 102–113. Springer-Verlag, LNBIP, Springer-Verlag, LNBIP, May 2009.

6 Fong lok Lee and Rex Heyworth. Problem complexity: A measure of problem difficulty in algebra by using computer, 2000.

7 Ricardo Queirós and José Paulo Leal. Petcha - a programming exercises teaching assistant. In *ACM SIGCSE 17th Anual Conference on Innovation and Technology in Computer Science Education*, Haifa, Israel, July 2012 2012. ACM.

8 Ricardo Queirós and José Paulo Leal. Pexil: Programming exercises interoperability language. Conferência - XML: Aplicações e Tecnologias Associadas (XATA), 2011.

9 Atiq Siddiqui, Mehmood Khan, and Sohail Akhtar. Supply chain simulator: A scenario-based educational tool to enhance student learning. *Comput. Educ.*, 51(1):252–261, August 2008.

10 John Sweller and Graham Cooper. The Use of Worked Examples as a Substitute for Problem Solving in Learning Algebra. *Cognition and Instruction*, 2:59–89, 1985.

11 Maarten Vansteenkiste and Edward L. Deci. Competitively contingent rewards and intrinsic motivation: Can losers remain motivated? *Motivation and Emotion*, 27:273–299, 2003. 10.1023/A:1026259005264.

12 Fu Lee Wang and Tak-Lam Wong. Designing programming exercises with computer assisted instruction. In *Proceedings of the 1st international conference on Hybrid Learning and Education*, ICHL '08, pages 283–293, Berlin, Heidelberg, 2008. Springer-Verlag.

13 Susan Wiedenbeck, Deborah Labelle, and Vennila N. R. Kain. Factors affecting course outcomes in introductory programming. In *In 16th Annual Workshop of the Psychology of Programming Interest Group*, pages 97–109, 2004.