Design a Computer Programming Learning Environment for Massive Open Online Courses

Ricardo Queirós
ESEIG/IPP, Polytechnic Institute of Porto, Portugal & CRACS/INESC TEC

ABSTRACT

Teaching and learning computer programming is as challenging as difficult. Assessing the work of students and providing individualised feedback to all is time-consuming and error prone for teachers and frequently involves a time delay. The existent tools and specifications prove to be insufficient in complex evaluation domains where there is a greater need to practice. At the same time Massive Open Online Courses (MOOC) are appearing revealing a new way of learning, more dynamic and more accessible. However this new paradigm raises serious questions regarding the monitoring of student progress and its timely feedback.

This paper provides a conceptual design model for a computer programming learning environment. This environment uses the portal interface design model gathering information from a network of services such as repositories and program evaluators. The design model includes also the integration with learning management systems, a central piece in the MOOC realm, endowing the model with characteristics such as scalability, collaboration and interoperability.

This model is not limited to the domain of computer programming and can be adapted to any complex area that requires systematic evaluation with immediate feedback.

Keywords: Teaching Assistant, Automatic Evaluation, Programming Exercises, Interoperability, Learning Objects, MOOC.

INTRODUCTION

The evolution of e-learning in the last decades has been astonishing. In fact, e-learning seems to be constantly reinventing itself, finding new uses for technology, creating new tools, discovering new concepts. Platforms for supporting e-learning have been evolving for some years, exploring many approaches and producing a great variety of solutions.

These solutions make the learning and teaching more efficient and productive, but they usually lack effective real-time monitoring to learning process (Henda, 2013).

In the meantime many universities and institutions are using platforms for Massive Online Open Courses (MOOCs), characterised with a great diversity of topics and a huge number of enrolments. However, the real-time feedback is important for the effectiveness of MOOCs. We state that novice students in an e-learning system might feel being isolated from the teachers and other students, because of the lack of essential interactions components in the system design (Jonas & Burns, 2013). This issue leads to a negative impact on the students' outcome. With well-designed synchronous virtual classrooms and collaborative tools it is possible to reduce this negative impact (Nedeva & Dineva, 2010).

This issue augments when we talk about complex domains. Learning complex skills is hard. A good example is the computer programming domain. Introductory programming courses are generally regarded as difficult and often have high failure and dropout rates (Ala-Mutka, 2005), (O'Kelly & Gibson, 2006) and (Robins et al, 2003). Many educators claim that "learning through practice" is by far the best way to learn computer programming and to engage novice students (Jonas & Burns, 2013), (Eckerdal, 2009). Practice in this area boils down to solving programming exercises. Nevertheless, solving exercises is only effective if students receive an assessment on their work. Assessing the work of students and providing individualised feedback to all students is time-consuming for teachers and frequently involves a time delay. The existent tools and specifications prove to be insufficient in complex evaluation domains where there is a greater need to practice (Rongas & Kaarna, 2004).

This paper presents a conceptual design model for learning environments regarding complex domains. Specifically, we focus on the computer programming domain. This environment uses the portal interface design model gathering information from a network of services such as repositories and program evaluators. These services will improve the responsiveness of the environment, a crucial success factor in massive courses.

The design model includes also the integration with learning management systems, a central piece in the MOOC realm, endowing this way the model with characteristics such as scalability, collaboration and interoperability.

The remainder of this paper is organised as follows: the next section presents a brief survey on integration specifications such as, the digital repositories interoperability specification and the learning tools interoperability specification. Next, we present the conceptual model of a learning environment for a complex domain such as the computer programming domain. In the following section we propose a graphical user interface for such model focusing on the user profiles and actions, screen layout and implementation details. Finally, we conclude with a summary of the main contribution of this work and a perspective of future work.

INTEGRATION SPECIFICATIONS

The current generation of e-learning platforms values the interchange of learning objects and learners' information through the adoption of standards that brought content sharing and interoperability to eLearning. Learning Objects (LO) are units of instructional content that can be used, and most of all reused, on web based eLearning systems. Despite its success in the promotion of the standardization of eLearning content, it is not enough to ensure interoperability, which is a major user concern with the existing systems. The definition of common protocols and interfaces for the communication among systems is also an important issue to address.

In the last few years there have been initiatives (Leal & Queirós, 2010) to adapt Service Oriented Architectures (SOA) to e-learning. These initiatives, commonly named e-learning frameworks, had the same goal: to provide flexible learning environments for learners worldwide. Usually they are characterized by providing a set of open interfaces to numerous reusable services organized in genres or layers and combined in service usage models.

While eLearning frameworks are general approaches for e-learning system integration, several authors proposed service oriented approaches specifically targeted to the LMS. In fact, there are several

references in the literature to middleware components for LMSs integration in SOA based eLearning systems. Apostolopoulos proposes a middleware component (Apostolopoulos & Kefala, 2003) to address the lack of integration of eLearning services. In this approach the eLearning components are implemented as agents maintained in a local management information base, and can communicate with the agent manager through the SNMP protocol. Costagliola develop an architecture (Casella et al, 2007) based on a middleware component and use Web Services to integrate different software components and improve interoperability among different systems. The middleware component enables the student learning process traceability since it has been developed to be compliant with SCORM. Al- Smadi presents a service-oriented architecture (Al-Smadi & Gutl, 2010) for a generic and flexible assessment system with cross-domain use cases. All these approaches have in common the need of a modification of LMS for each specific vendor, with the implementation of a new module or building block. To the best of the authors' knowledge there are no references in the literature to the use of a common standards supported by the major LMS vendors as a means to integrate the LMS in a service oriented network of learning environments.

Other e-learning interoperabiliy initiatives (for instance, NSDL, POOL, OKI, EduSource, IMS DRI, IMS LTI) appeared in the last. We detail the last two ones in the following subsections.

Digital Repositories Interoperability

The IMS DRI provides recommendations for common repository functions, namely the submission search and download of LOs. It recommends exposing these functions as SOAP web services. Although not explicitly recommended, other web service interfaces may be used, such as the Representational State Transfer (REST). SOAP web services are usually action oriented, especially when used in Remote Procedure Call (RPC) mode and implemented by an off-the-shelf SOAP engine such as Axis. REST web services are object (resource) oriented and implemented directly over the HTTP protocol, mostly to put and get resources. The reason to provide two distinct web service flavours is to encourage the use of the repository by developers with different interoperability requirements. A system requiring a formal an explicit definition of the API in Web Services Description Language, to use automated tools to create stubs, will select the SOAP flavour. A lightweight system seeking a small memory footprint at the expense of a less formal definition of the API will select the REST flavour. The following paragraphs detail the main functions.

The **Submit/Store** function uploads an LO to a repository and makes it available for future access. This operation receives as argument an IMS CP compliant file and an URL generated by the Reserve function. This operation validates the LO conformity to the IMS Package Conformance and stores the LO in the internal database. To send the LO to the server we could use, in the REST flavour, the PUT or the POST HTTP methods.

The **Search/Expose** function enables the eLearning systems to query the repository using the XQuery language, as recommended by the IMS DRI. This approach gives more flexibility to the client systems to perform any queries supported by the repository's data. To write queries in XQuery the programmers of the client systems need to know the repository's database schema. These queries are

based on both the LO manifest and its usage reports, and can combine the two document types. The client developer needs also to know that the database is
structured in collections. A collection is a kind of a folder containing several resources and sub-folders. From the XQuery point of view the database is a collection of manifest files. For each manifest file there is a nested collection containing the usage reports. As an example of a simple search, suppose you want to find all the titles of LOs in the root collection whose author is Manzoor. The XQuery file would contain the data.

The **Report/Store** function associates a usage report to an existing LO. This function is invoked by the LMS to submit a final report, summarizing the use of an LO by a single student. This report includes both general data on the student's attempt to solve the programming exercise (e.g. data, number of evaluations, success) and particular data on the student's characteristics (e.g. gender, age, instructional level). With this data, the LMS will be able to dynamically generate presentation orders based on previous uses of LO, instead of fixed presentation orders. This function is an extension of the IMS DRI.

The **Alert/Expose** function notifies users of changes in the state of the repository using a RSS feed. With this option a user can have up-to-date information through a feed reader. Next, we present an example of a GET HTTP request.

Learning Tools Interoperability

A common interoperability standard that is increasingly supported by major LMS vendors is the IMS Learning Tools Interoperability (IMS LTI) specification. It provides a uniform standards-based extension point in LMS allowing remote tools and content to be integrated into the LMS. The main goal of the LTI is to standardize the process for 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, increases options for students and instructors when selecting learning applications and also potentiates the use of software as a service (SaaS).
The LTI has 3 key concepts as shown in Figure 1 (Gilbert, 2010): the Tool Provider, the Tool Consumer and the Tool Profile.

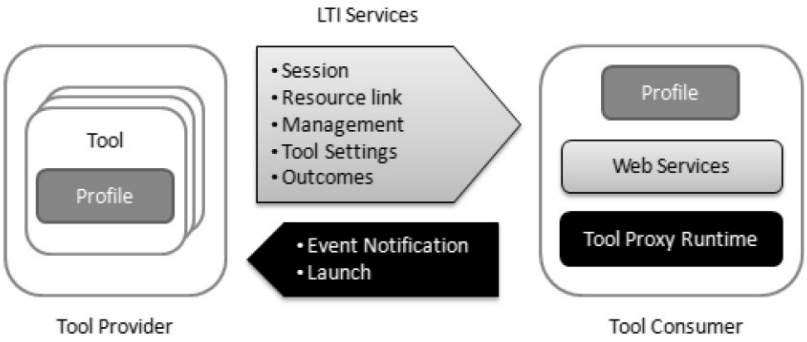

Figure 1 *– The IMS LTI framework.*

The Tool Provider is a learning application that runs in a container separate from the LMS. It publishes one or more tools through the Tool Profiles. The Tool Profile is an XML descriptor that describes how a tool integrates with a tool consumer. It is composed by information about the 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.
The IMS launched also a subset of the full LTI v1.0 specification called IMS Basic LTI. This subset exposes a single (but limited) destination between the LMS and the application as shown in Figure 1.
For instance, there is no provision for accessing run-time services in the LMS and only one security policy is supported. Basic LTI also supports a basic security model based on the OAuth protocol. This protocol aims to secure the message interactions between the Tool Consumer and the Tool Provider. It requires a key and a shared secret to sign messages. The key is sent with each message, as well as an OAuth-generated signature based on the key. The Tool Provider verifies the secret based on the provided key and re-computes the signature and compares the recomputed signature with the transmitted signature to verify the sender's credentials.

CONCEPTUAL MODEL

Typically, a conceptual model represents entities and relationships between them regarding a specific domain. Therefore, we present the conceptual model for the design of a computer programming teaching/learning environment. The aim of this conceptual model is to express the meaning of domain concepts and the correct relationships between different concepts. The model for the computer programming learning environment (CP-LE) is depicted by the UML component diagram in Figure 2 composed by the following concepts:

- Learning Objects Repository (LOR) to store/retrieve exercises;
- Assessment System (AS) to evaluate students exercises
- Learning Management System (LMS) to present the exercises to students;
- Converter System (CS) to convert between different exercise formats;

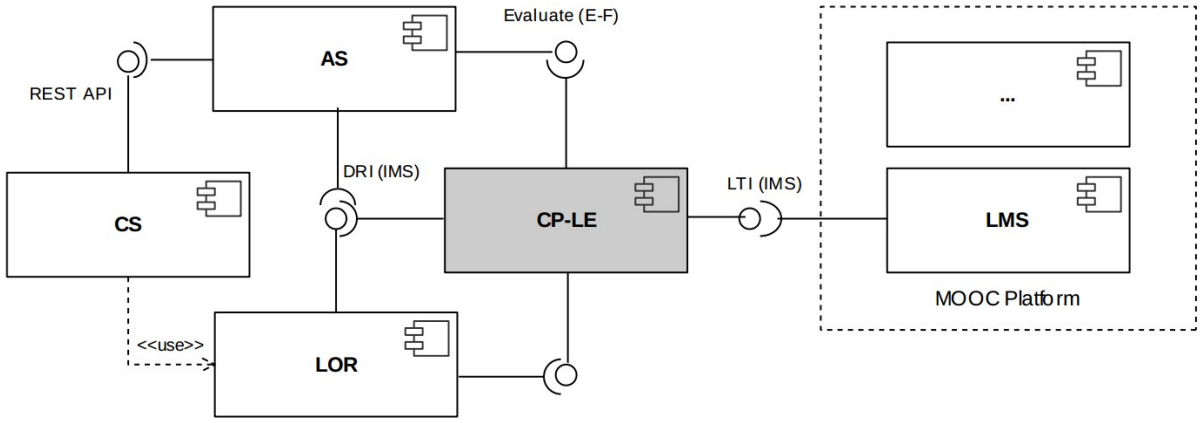

Figure 2 – *Conceptual Model.*

The CP-LE has a two-fold goal: to coordinate the systems and services of this network and to interface with users, both teachers and students. On the LMS side the choice fell on Moodle since it is a popular and open source LMS, arguably the most popular LMS nowadays (Cole & Foster, 2007), (Davis & Wagner, 2009). This LMS has made efforts to support interoperability with other e-learning systems at two levels: content (e.g. IMS CP, SCORM, IMS CC) and communication (e.g. IMS LTI). Also successfully tests were made with Sakai LMS on this network evidencing the interoperable characteristics of the proposed approach.

The LOR system selected was CrimsonHex (Queirós & Leal, 2013) - a software for the creation of repositories of programming exercises. The exercises are described as learning objects and complying with the IMS CC specification. The repository also adheres to the IMS DRI specification to communicate with other systems. Other software for repositories were analysed (e.g. Flori, HarvestRoad Hive, IntraLibrary) but none of them met the domain requirements for the content and communication interoperability and most of them follow a commercial development model.

The AS system selected was Mooshak (Leal & Silva, 2003). Mooshak is an open source system for managing programming contests on the Web including automatic judging of submitted programs. One of the most important reasons for its selection was the support of web services.

The CS system selected was BabeLO (Queirós & Leal, 2013). This system converts formats of programming exercises among systems. At the time of writing this dissertation no other system was found with these characteristics.

The integration of the CP-LE component with the other systems must rely on content and communication standards. Using content and communication standards we can abstract the use of specific systems for each type of system. For instance, we can use on this network any repository as long it supports the IMS CC specification to formalize the description of programming exercises and it implements the IMS DRI specification for communication with other services.

In this particular scenario the teacher starts by setting a number of activities in the LMS, including the resolution of programming exercises. To select the relevant programming exercises the teacher 1) searches for relevant exercises in the repository. Then, the learner 2) tries to solve the exercises set be the teacher using an Experimentation Environment (e.g. Eclipse IDE). The IDE 3) recovers exercises descriptions from the repository showing them to the student. After coding the program the learner 4) send an attempt to the Evaluation Engine. The Evaluation Engine 5) recovers test cases from the repository. The learner may submit repeatedly, integrating the feedback received from the Evaluation Engine. In the end, the Evaluation Engine 6) sends a grade to the LMS that records it and reports the LO usage data back to the repository.

Repositories

Learning objects repositories are an essential part of service oriented platforms in eLearning since they provide content to several types of services. The need for this kind of repositories is growing as more educators are eager to use digital educational contents and more of it is available. Several surveys show that users are concerned with issues that are not completely addressed by the existing systems, such as interoperability. Thus, a desired feature of a repository is the support for a standard and automatic communication with other systems.
The repository used in this work was the crimsonHex. It was developed as part of the EduJudge project (Leal & Queirós, 2010) to act as a programming problem repository service. The Core component of the repository exposes the main features, both to external services, such as the LMS and the AS, and to internal components, respectively: the Web Manager, to allow the creation, revision, uploading/downloading of LO and related metadata, enforcing compliance with controlled vocabularies; and the Importer, to populate the repository with existing legacy repositories.

The Core component of the crimsonHex repository provides a minimal set of operations exposed as web services – in SOAP and REST flavours - and based in the IMS DRI specification.

Assessment Systems

The purpose of a programming exercise evaluator is to mark and grade exercises in computer programming courses and in programming contests. By exposing its functions as services, an evaluator of this kind is able to participate in business processes integrating different system types such as Programming Contest Management Systems, Learning Management ystems, Integrated Development Environments and Repositories.

In order to formalize the definition of this service we used an eLearning framework. An eLearning framework aims to adapt SOA to eLearning providing flexible learning environments for learners worldwide. The new service - Evaluate Programming Exercise - models the evaluation of an attempt to solve a programming exercise defined as a learning object and produces a detailed report. This evaluation report includes information to support exercise assessment, grading and/or ranking by client systems. This service exposes its functions as SOAP and REST web services. The three types of request handled by this service are:

- ListCapabilities - provides the client systems with the capabilities of a particular evaluator;
- EvaluateSubmission - allows the request of an evaluation for a specific programming exercise;
- GetReport - allows a requester to get a report for a specific evaluation using a ticket.

More details about the definition of this service can be found elsewhere (Leal & Queirós, 2010).

Integrated Development Environments

Experimenting environments – environments for practicing on a learning subject to consolidate learning – are another type of specialized services to be integrated in learning processes. These environments need a user interface to interact with learners and application interfaces to be integrated on

the learning process. In some cases they will have to be developed for specific domain, while in other they can be adapted from existing systems.

Take the computer language programming domain as an example. An Integrated Development Environment (IDE) is arguably the best place for a student to practice by solving programming exercises, but any tool on a CLMS will hardly be a match. Surely, an IDE lacks the features to communicate with other specialized services, but this shortcoming may be overcome using plugins, similar to those described in the previous sub-section for Moodle.

One approach is to use rich web editors such as Ace. Ace is an embeddable code editor written in JavaScript. It matches the features and performance of native editors such as Sublime, Vim and TextMate. It can be easily embedded in any web page and JavaScript application. Ace is maintained as the primary editor for Cloud9 IDE and is the successor of the Mozilla Skywriter (Bespin) project.

A GUI PROPOSAL

In this section we propose a possible GUI for the learning environment. In the design of the Web component one of our major concerns was usability, and to promote it we followed established user interface design principles (Shneiderman, 1998). The main feature of the resulting design is the use of a single screen common to all user profiles. This type of design breaks with the traditional structure of web interfaces used by other systems (A. Co-Lab, 2003). To design this user interface we started with the identification of task and usage profiles, task objects and task actions. Then we selected a suitable interaction style and finally we created a screen layout.

User Profiles and Actions

At the beginning of the design process we identified the following task profiles:

- Administrator - a person responsible for the management of the system configurations such as user accounts and repository settings;
- Teacher - a person responsible for a set of activities related with the resource management such as the authoring of two type of resources: expository (e.g. video, PDF or HTML files) and evaluation resources (programming exercises) and the submission of the resources in the repository. The submission will be enforced to comply with controlled vocabularies defined in meta-data standards (IEEE LOM) and possible extensions. This class of users will also receive the exercises solved by students and the automatic feedback generated by the assessment system;
- Student - a person that browses the resources and solve exercises.

We assume that users will have different usage profiles. On one hand, many will be novice or first-time users, especially among students. On the other hand, we expect some users, especially teachers and old students, to use it frequently, tending to become experts in its use. After the identification of users and usage profiles we proceeded to identify the tasks they need to perform on this interface. We clearly identified expository and evaluation resources as our task objects, each with a number of associated task

actions, depending on user profiles. Task actions over resources include: viewing, downloading, solving, voting, sharing and commenting.

A typical pedagogical learning process is the classroom assignment in a Computer Science course. For instance, when a student starts solving an exercise, the CP-LE component automatically creates a project. A project contains source code and related files for building a program in a specific programming language. Thus, a set of predefined files need to be generated for the project creation. These files are related with the chosen programming language.

After the automatic creation of the project the student reads the exercise description and solves it in a specialized Web editor (e.g. AceEditor). The student should test the code locally by executing the teachers' test cases and is encouraged to create new ones. If new test cases are created, a validation step is performed to verify that they meet the specification defined by the teacher in the authoring phase. After testing, the student should submit the solution to the Assessment System where the submission is checked against the complete test set provided by the teacher. The report on the evaluation returned by the AS is presented to the student. The student may submit repeatedly, integrating the feedback received from the AS. In the end of this cycle, the CP-LE component reports the exercise usage data back to the repository and the grade results back to the LMS.

Screen Layout

To define the screen layout we sought an interaction style balancing intuitiveness and expressiveness. We first considered direct manipulation of task objects. Although it provides a convenient way to select objects, it is not possible to map all the identified task actions to basic mouse interaction (click, point and drag). We found form filling adequate for entering data for the complex tasks, such as search, commenting and solving. Finally, we decided to blend these two interaction styles, using a form of direct manipulation for task object selection and form-filling for executing task actions.
Based on this blended interaction style we defined a screen layout - a single screen with specific areas for task object selection and task actions. Task object selection is needed by all users, although selectable content depends on the user's profile, thus it can be implemented by a common tree-based control. Different task actions require specific forms or panels that also share a common control on the users interface. Since the number of task actions is comparatively small we chose a tabbed control to aggregate them. The tab configuration shown to users depends both on their profile and on the current task object selection.

Figure 3 shows the user interface layout of the computer programming learning environment with two main areas: selection on the left side and action on middle. In the selection area the user navigates through the repository structure to select task objects. In the action area the user executes task actions (e.g. view videos, solve exercises) on the selected task objects. Secondary areas in this layout are the header, used for authentication and registration, and the right side, used for statistics and chat.
As a rule, all available task actions are enabled, thus helping novice users to recognise which are the available actions. However, some of these task actions can be executed directly over selected task objects without requiring additional data. In general, these task actions are meant for frequent users and will be bound to contextual menus on the tree-control, as well as to accelerator keys.

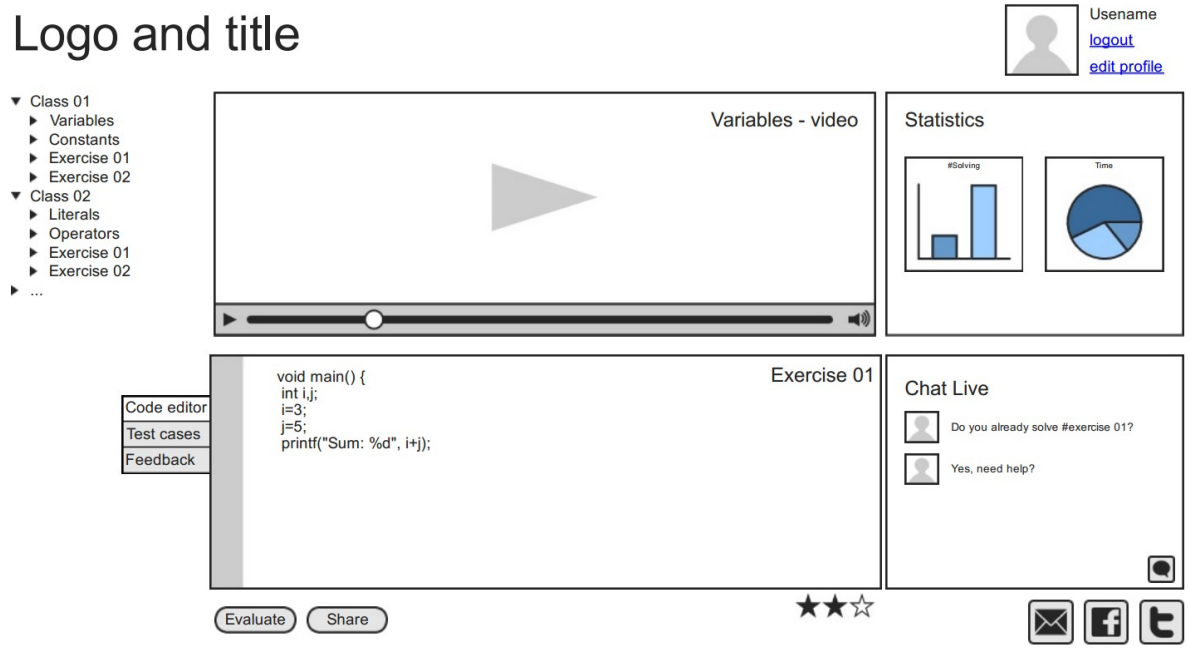

Figure 3 *– Screen Layout.*

Implementation

The learning environment was developed using an Ajax framework to enable the implementation of the single screen design resulting from the last section. We selected the Google Web Toolkit (GWT), 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 framework supports also asynchronous remote procedure calls. This way, tasks that require high computational resources (e.g., complex searching within the repository) can be triggered asynchronously, increasing the user interface's responsiveness. The complex controls required by the selection and action areas are provided by SmartGWT, a GWT API's for SmartClient, a Rich Internet Application (RIA) system.

The Web component 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 rely on the gateway class for managing the communication with the Web services.
A single class implementing the Gateway design pattern concentrates the interaction with the core component.
The integration of the pivot component in the LMS relies on the LTI specification. The basic workflow for using Basic LTI starts when the Teacher (or LMS administrator) adds the tool as a Basic LTI tool into their course structure as a resource link using the LMS control panel. The Teacher sets the URL, secret, and key as metadata for the resource link. When the students select the tool, the LMS uses the URL, secret, and key information to launch the student into the CP-LE in an iframe or new browser window.

The CP-LE component receives a launch request that includes user identity, course information, role information, and the key and signature. The launch information is sent using an HTTP form generated in the user's browser with the Basic LTI data elements in hidden form fields and automatically submitted to the external tool using JavaScript. The following is a subset of the information that the LMS (Tool Consumer) sends to the CP-LE (Tool Provider):

```
resource_link_id=1 //An unique identifier for the resource in the LMS.
resource_link_title= My First Exercise // Title of the resource
resource_link_description= Description... //Description of the resource.
user_id=2 // User identifier
user_image=myPhoto.gif // Profile picture
roles= Instructor,Administrator //List of one or more user roles.
context_title=Course Fullname 101 //A title of the context (e.g. course information).
```

All these data items are included on the POST data when a Basic LTI launch is performed. These data items can be used, for instance, to personalize the frontend of the tool providers. To extend these fields it is necessary to prefix all fields not described herein with "ext_".

CONCLUSIONS

This paper presents a conceptual model for the design of a learning environment for the teaching and learning process in complex domains such as computer programming. The design model is suitable for integration in MOOC platforms where there are a large number of enrolments and, at the same time, a large number of dropouts due to lack of teacher support. The adaptation to MOOC platforms is guaranteed with the integration of systems that provide automatic assessment giving to the student a higher autonomy to proceed in the course without the need to wait for teachers' feedback.

The model could be adapted to other complex domains. Playing business games in management courses or simulating a human patient in life sciences courses, or simulating an electronic circuit in electronics courses are examples of complex learning domains that require the use of special evaluators. Currently we have Petcha (a CP-LE component) running at ESEIG - an Engineering School - with promising results.

Regarding future work we expected to include other services in this network with the inclusion of a plagiarism tool to avoid plagiarism and ensure good scholarly practices and a resources sequencing tool. Sequencing of exercises is another topic that can be explored in the future and it is closely related with pedagogical issues during the construction of a learning scenario. Several standards appeared in recent years trying to cope this topic but fail due its complexity for e-Learning systems to implement. One research path is to deliver exercises to students dynamically according with their profiles, knowledge evolution and course goals. An intended addition is a sequencing and adaptation tool to guide the student through a collection of expository and evaluation resources. The CP-LE component will report the exercise assessment to this new tool that will use it to propose the appropriate content or exercise to the student.

We concluded that a pivot component integrated in the LMS is a promising approach to the task of coordinating a heterogeneous network of e-learning systems. The pivot component can have its own

user interface for interaction with students, as is required for the resolution environment, that is embed in the LMS user interface. It can also control the invocation of remote web services, such as those exposed by the repository of learning objects and evaluation engine. Finally, it can summarize the activity of the student as a grade and report it back to the grade book of the LMS.
Unfortunately, we must conclude also that the LMS support of LTI standard in not mature enough for using this approach in the near future. Most LMS vendors, and in particular those we tested, support only the Basic LTI. Thus, the grade reporting feature could not be fully implemented in our integration and validation setup. Moreover, we had to perform custom installations of both LMSs, mixing code from a stable distribution and code under development just to have Basic LTI.

A full and stable support of LTI in major LMS vendors will encourage us to implement a more sophisticated version of the approach described in this paper. Instead of embedding the resolution environment on the LMS the student should be able to use and Integrated Development Environment (IDE) such as Eclipse. We plan to develop an IDE plug-in to create a programming exercise resolutions environment. It will complement the standard code programming features of an IDE with reading exercise descriptions from the repository, submitting code them to the evaluation engine and displaying feedback to the student. In this future work we will split the coordination task among the pivot component integrated in the LMS and the plug-in on the IDE. Also, the LMS must communicate with a local service on the student's machine (hosted on the IDE) rather than on the cloud. Still, this variant is a step towards to integrate in this network the best of bread for each task.

REFERENCES

A. Co-Lab (2003). Learning repositories included in learning repository investigation. Retrieved from http://www.academiccolab.org/resources/DraftRepositoriesList.pdf.

Al-Smadi, M.; Gutl, C. (2010). SOA-based architecture for a generic and flexible e-assessment system. In: EDUCON.

Ala-Mutka, K. (2005). A survey of automated assessment approaches for programming assignments. Journal of Computer Science Education, 15 (2), 83-102.

Apostolopoulos, T.K., Kefala, A.S. (2003). An e-learning service management architecture. In: ICALT. pp. 140–144.

Casella, G., Costagliola, G., Ferrucci, F., Polese, G., Scanniello, G. (2007). A SCORM thin client architecture for e-learning systems based on web services. IJDET 5(1), 19–36.

Cole, J., & Foster, H. (2007). Using Moodle: Teaching with the Popular Open Source Course Management System (2nd ed.). O'Reilly Media. Paperback. Retrieved from http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/059652918X
Davis, C. C., B., & Wagner, E. (2009). The Evolution of the LMS: From Management to Learning - Deep Analysis of Trends Shaping the Future of eLearning (Tech. Rep.). Sage Road Solutions, LLC.

Eckerdal, A. (2009). Novice programming students' learning of concepts and practise (Unpublished doctoral dissertation). Uppsala University Uppsala University, Division of Scientific Computing, Numerical Analysis.

Gilbert, T. (2010). Leveraging sakai and ims lti to standardize integrations. In 10th sakai conference.

Henda, Belaid-Ajroud (2013) .Monitoring activities in an e-learning 2.0 environment: A multi-agents system. The Eighth International Conference on Internet and Web Applications and Services, ICIW.

Jonas, N. C. and Burns, Denise (2013). Using e-learning to educate health professionals in the management of children's pain.6th International Conference Creativity Engagement in Higher Education.

Leal, J. P., & Silva, F. M. A. (2003). Mooshak: a web-based multi-site programming contest system. Softw., Pract. Exper., 33 (6), 567-581.

Leal, J.P. & Queirós R. (2010). From eLearning Systems to Specialised Services. Chapter of EduJudge project book called "A New Learning Paradigm: Competition Supported by Technology". Sello Editorial.

Nedeva, E. D. and Dineva, S. (2010). Overcome disadvantages of e-learning for training English as foreign language. ICVL 2010: Proceedings of The 5thInternational Conference on Virtual Learning.

O'Kelly, J. and Gibson, J. P. (2006). Robocode & problem-based learning: a non-prescriptive approach to teaching programming. SIGCSE Bull., 38(3):217-221.

Queirós, R. & Leal, J. P. (2013). crimsonHex: a learning objects repository for programming exercises.. Softw., Pract. Exper., 43, 911-935.

Queirós, R. & Leal, J. P. (2013). BabeLO - An Extensible Converter of Programming Exercises Formats, IEEE Transactions on Learning Technologies, vol. 6, no. 1, pp. 38-45.

Robins, A., Rountree, J. & Rountree, N. (2003). Learning and teaching programming: A review and discussion. Computer Science Education, 13 , 137-172.

Rongas, T. and Kaarna, A. (2004). Classification of computerized learning tools for introductory programming courses: Learning approach. In Kinshuk et al. (Eds.), ICALT. IEEE Computer Society. Retrieved from http://dblp.uni-trier.de/db/conf/icalt/icalt2004.htmlRongasKK04

Shneiderman, B. (1998) Designing the Users Interface -Strategies for Effective Human-Computer Interaction.Addison-Wesley, 3 edition.

ADDITIONAL READING

Alario, C., & Wilson, S. (2010, 15-17 November, 2010). Comparison of the main alternatives to the integration of external tools in different platforms. In Iceri2010 proceedings (p. 3466-3476). IATED.

Al-Khalifa, H. S., & Davis, H. C. (2006). The evolution of metadata from standards to semantics in e-learning applications. In Hypertext (p. 69-72). (http://doi.acm.org/10.1145/1149941.1149956)

Aroyo, L., Dolog, P., jan Houben, G., Kravcik, M., Naeve, A., Nilsson, M., & Wild, F. (2006). Interoperability in personalized adaptive learning. Journal of Educational Technology & Society, 9 (2), 4-18. (http://www.ifets.info/journals/92=2:pdf)

Ashford-Rowe, K., & Malfroy, J. (n.d.). E-learning benchmark report: Learning management system (lms) usage (Tech. Rep.). Sydney.

Barker, P., & Campbell, L. M. (2010). Metadata for learning materials: an overview of existing standards and current developments. Technology, Instruction, Cognition and Learning, 7(3-4), 225-243. (http://www.icbl.hw.ac.uk/publicationFiles/2010/TICLMetadata/TICLpaper.MetadataForEducationpostref:pdfBarret, H. (n.d.). Categories of eportfolio tools (Tech. Rep.). JISC.

Barret, H. (2010). Electronic portfolios in stem - what is an electronic portfolio. (http://www.scribd.com/doc/40206175/E-Portfolio-Definition)

Benford, S., Burke, E., Foxley, E., Gutteridge, N., & Zin, A. (2011). Early experiences of computer-aided assessment and administration when teaching computer programming. Research in Learning Technology, 1 (2). Retrieved from http://www.researchinlearningtechnology.net/index.php/rlt/article/view/9481

Bersin, H. C. O. K. M. D., J. (2009). Learning management systems 2009: Executive summary. Bersin & Associates.

Blumenstein, M., Green, S., Nguyen, A., & Muthukkumarasamy, V. (2004, June). An experimental analysis of game: a generic automated marking environment. SIGCSE Bull., 36 , 67-71. Retrieved from http://doi.acm.org/10.1145/1026487.1008016

Borg, W. R., & Gall, M. D. (2007). Educational research; an introduction, by walter r. borg and meredith d. gall (8th ed. ed.) [Book]. McKay New York.

Britain, L. O., S. (1998). A Framework for Pedagogical Evaluation of Virtual Learning Environments (Tech. Rep.). (http://www.leeds.ac.uk/educol/documents/00001237.htm)

Burguillo, J. C. (2010, September). Using game theory and competition-based learning to stimulate student motivation and performance. Comput. Educ., 55 (2), 566-575. Retrieved from http://dx.doi.org/10.1016/j.compedu.2010.02.018 doi: 10.1016/j.compedu.2010.02.018

Chae, G. B., Chandra, S., Mann, V., & Nanda, M. G. (2004). Decentralized orchestration of composite web services. In Proceedings of the 13th international world wide web conference on alternate track papers & posters (pp. 134-143). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/1013367.1013390 doi: 10.1145/1013367.1013390

Cheang, B., Kurnia, A., Lim, A., & Oon, W.-C. (2003, September). On automated grading of programming assignments in an academic institution. Comput. Educ., 41 , 121-131. Retrieved from http://dl.acm.org/citation.cfm?id=941987.941991 doi: 10.1016/S0360-1315(03)00030-7

Chloros, G., Zervas, P., & Sampson, D. G. (2010). Ask-lom-ap: A web-based tool for development and management of ieee lom application profiles. In Icalt (p. 138-142). (http://dx.doi.org/10.1109/ICALT.2010.46)

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., & Weerawarana, S. (2002, March). Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. Internet Computing, IEEE, 6 (2), 86-93. (http://dx.doi.org/10.1109/4236.991449) doi: 10.1109/4236.991449

Dagger, D., O'Connor, A., Lawless, S., Walsh, E., & Wade, V. P. (2007). Service-Oriented E-Learning Platforms: From Monolithic Systems to Flexible Services. Internet Computing, IEEE, 11 (3), 28-35. (http://ieeexplore.ieee.org/xpls/abs/all:jsp?arnumber =4196172)

Daly, C. (1999, June). Roboprof and an introductory computer programming course. SIGCSE Bull., 31 (3), 155-158. Retrieved from http://doi.acm.org/10.1145/384267.305904 doi: 10.1145/384267.305904

Donello, J. (2002). Theory & practice: Learning content management systems. ELearningMag.
Douce, C., Livingstone, D., & Orwell, J. (2005, September). Automatic test based assessment of programming: A review. J. Educ. Resour. Comput. 5 . Retrieved from http://doi.acm.org/10.1145/1163405.1163409 doi: http://doi.acm.org/10.1145/1163405.1163409

Eap, T. M., Hatala, M., & Richards, G. (2004). Digital repository interoperability: design, implementation and deployment of the ecl protocol and connecting middleware. In Proceedings of the 13th international world wide web conference on alternate track papers & posters (pp. 376-377). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/1013367.1013483 doi: 10.1145/1013367.1013483

Eckerson, W. W. (1995). Three tier client/server architecture: Achieving scalability, performance, and efficiency in client server applications. Open Information Systems, 10 (1).

Edwards, S. H., Borstler, J., Cassel, L. N., Hall, M. S., & Hollingsworth, J. (2008). Developing a common format for sharing programming assignments. SIGCSE Bull., 40 (4), 167-182. Retrieved from http://dx.doi.org/10.1145/1473195.1473240 doi: 10.1145/1473195.1473240

Edwards, S. H., & Pugh, W. (2006, March). Toward a common automated grading platform. In Birds-of-a-feather session at the 37th sigcse technical symposium on computer science education.

Ellis, R. K. (2009). Field guide to learning management systems. ASTD Learning Circuits.

Engels, S., Lakshmanan, V., & Craig, M. (2007, March). Plagiarism detection using feature-based neural networks. SIGCSE Bull., 39 , 34-38. Retrieved from http://doi.acm.org/10.1145/1227504.1227324 doi: http://doi.acm.org/10.1145/1227504.1227324

Erl, T. (2005). Service-oriented architecture - concepts, technology and design. Prentice Hall.

Esteves, M., Fonseca, B., Morgado, L., & Martins, P. (2010, March). Improving teaching and learning of computer programming through the use of the Second Life virtual world. British Journal of Educational Technology. Retrieved from http://dx.doi.org/10.1111/j.1467-8535.2010.01056.x
doi: 10.1111/j.1467-8535.2010.01056.x

Farance, F., & Tonkel, J. (1999). Ltsa specification - learning technology systems architecture, draft 5 (Tech. Rep.). IEEE. Retrieved fromhttp://ltsc.ieee.org/wg1/files/ltsa05:pdf

Fay, E. (2010). Repository software comparison: Building digital library infrastructure at lse. Ariadne, 64 . (http://www.ariadne.ac.uk/issue64/fay/)

Fernandez, J. L., Carrillo, J. M., Nicolas, J., Toval, A., & Carrion, M. I. (2011). Trends in e-learning standards. International Journal of Computer Applications, 353 (1), 49-54. Retrieved from http://www.ijcaonline.org/dedce/number1/dece008.pdf

Fielding, R., & Taylor, R. (2000). Principled design of the modern web architecture. In Software engineering, 2000. proceedings of the 2000 international conference on (p. 407 -416). doi: 10.1109/ICSE.2000.870431

Friesen, N. (2004a). In Metadata in practice (chap. Semantic and Syntactic Interoperability for Learning Object Metadata). ALA Editions.

Friesen, N. (2004b). Editorial - a gentle introduction to technical elearning standards. Canadian Journal of Learning and Technology, 30 (3). (http://www.cjlt.ca/index.php/cjlt/article/view/136)

Friesen, N. (2005). Interoperability and learning objects: An overview of e-learning standardization. Interdisciplinary Journal of Knowledge and Learning Objects.

Gomes, A., & Mendes, A. J. (2007). Learning to program - difficulties and solutions. Proceedings of the International Conference on Engineering Education. Retrieved from http://icee2007.dei.uc.pt/proceedings/papers/411.pdf

Gray, L. (2008). Effective practice with e-portfolios: Supporting 21st century learning. JISC. (http://www.jisc.ac.uk/media/documents/publications/effectivepracticeeportfolios.pdf)

Gross, P., & Powers, K. (2005). Evaluating assessments of novice programming environments. In Proceedings of the first international workshop on computing education research (pp. 99-110). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/1089786.1089796 doi: 10.1145/1089786.1089796

Guerreiro, P., & Georgouli, K. (2008, 01). Enhancing elementary programming courses using e-learning with a competitive attitude. International Journal of Internet Education, 10 .

Gutierrez Rojas, I., Agea, , Crespo Garcia, R. M., Pardo, A., & Delgado Kloos, C. (2009). Assessment interoperability using qti. In Interactive conference on computer aided learning. Retrieved from http://www.iicm.tugraz.at/CAF2009

Harasim, L. (2006). A History of E-learning: Shift Happened. In (pp. 59-94). Retrieved from http://dx.doi.org/10.1007/978-1-4020-3803-7 2 doi: 10.1007/978-1-4020-3803-7 2

Harman, K., & Koohang, A. (2006). Learning objects: Standards, metadata, repositories, and LCMS. Santa Rosa, CA, USA: Informing Science Press. Higgins, C. A., Gray, G., Symeonidis, P., & Tsintsifas, A. (2005, September). Automated assessment and experiences of teaching programming. J. Educ. Resour. Comput., 5 . Retrieved from http://doi.acm.org/10.1145/1163405.1163410 doi: http://doi.acm.org/10.1145/1163405.1163410

Hoel, T., & Mason, J. (2011). Expanding the scope of metadata and the issue of quality. In 19th international conference on computers in education. (http://hoel.nu/publications/ICCEworkshop paper Hoel Mason2011/final.pdf)

Ihantola, P., Ahoniemi, T., Karavirta, V., & Seppala, O. (2010). Review of recent systems for automatic assessment of programming assignments. In Proceedings of the 10th koli calling international conference on computing education research (pp. 86-93). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/1930464.1930480

Jackson, D., & Usher, M. (1997). Grading student programming using assyst. In In technical symposium on computer science education, proceedings of the 28th sigcse (p. 335 -339).

Jena, S. (2008). Authoring and sharing of programming exercises (Unpublished master's thesis). San Jose State University. (http://scholarworks.sjsu.edu/etdprojects=19)

Jenkins, T. (2002). On the Difficulty of Learning to Program. In 3rd annual conference of ltsn-ics,. Loughbourgh. Retrieved from http://www.ics.ltsn.ac.uk/pub/conf2002/tjenkins.pdf

Jerman-Blazic, B., & Klobucar, T. (2005). Privacy provision in e-learning standardized systems: status and improvements. Computer Standards & Interfaces, 27 .(http://www.qou.edu/arabic/researchProgram/eLearningResearchs/privacyp.pdf)

Juedes, D. (2003). Experiences in web-based grading. In In 33rd asee/ieee frontiers in education conference (p. 5-8).

Kati Clements, J. M. P., Agueda Gras-Vel azquez. (n.d.). Educational resources packaging standards scorm and ims common cartridge - the users point of view. In Search and exchange of e-learning materials 2010 proceedings.

Klenin, A. (2011). Common problem description format: Requirements. ACMICPC World Final CLIS (Competitive Learning Institute Symposium).

Kumar, P., Samaddar, S., Samaddar, A., & Misra, A. (2010). Extending ieee ltsa e-learning framework in secured soa environment. In 2nd international conference on education technology and computer (icetc).

Kurilovas, E. (2012). European learning resource exchange: A platform for collaboration of researchers, policy makers, practitioners, and publishers to share digital learning resources and new e-learning practices. In . P. O. d. P. A. Cakir (Ed.), Social development and high technology industries: Strategies and applications. IGI-Global. Retrieved from http://www.igi-global.com/chapter/social-development-high-technology-industries/58723 doi: doi:10.4018/978-1-61350-192-4.ch014

Lahtinen, E., Ala-Mutka, K., & J arvinen, H.-M. (2005, June). A study of the difficulties of novice programmers. SIGCSE Bull., 37 (3), 14-18. Retrieved from http://doi.acm.org/10.1145/1151954.1067453 doi: 10.1145/1151954.1067453

Lawrence, A. W., Badre, A. M., & Stasko, J. T. (1994). Empirically evaluating the use of animations to teach algorithms. Visual Languages 1994 Proceedings IEEE Symposium on, pages, 48-54. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.25.8514rep=rep1type=pdf

Leal, J. P., & Queirós, R. (2012). A Comparative Study on LMS Interoperability. In I. Management Association (Ed.), Virtual Learning Environments: Concepts, Methodologies, Tools and Applications (pp. 1613-1630). Hershey, PA: Information Science Reference. doi:10.4018/978-1-4666-0011-9.ch804

Leal, J.P. & Queirós R. (2009), Defining Programming Problems as Learning Objects, Proceedings of International Conference on Computer Education and Instructional Technology, Venice, Italy.

Leslie, S. (2006). Challenges to Implementing DSpace as a LOR.

Levensaler, L., & Laurano, M. (2010). Talent management systems 2010: Market realities, implementation experiences and solution provider profiles. Bersin & Associates.

Liang, Y., Liu, Q., Xu, J., & Wang, D. (2009, dec.). The recent development of automated programming assessment. In Computational intelligence and software engineering, 2009. cise 2009. international conference on (p. 1 -5).(http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=arnumber=5365307) doi: 10.1109/CISE.2009.5365307

Luck, M., & Joy, M. (1999). A secure on-line submission system. In Software - practice and experience (pp. 721-740).

Malita, L. (2009). E-portfolios in an educational and ocupational context. Procedia - Social and Behavioral Sciences, 1 (1), 2312 - 2316. Retrieved from http://www.sciencedirect.com/science/article/pii/S1877042809004091 doi: 10.1016/j.sbspro.2009.01.406

Malmi, L., Karavirta, V., Korhonen, A., & Nikander, J. (2005, September). Experiences on automatically assessed algorithm simulation exercises with different resubmission policies. J. Educ. Resour. Comput., 5 Retrieved from http://doi.acm.org/10.1145/1163405.1163412 doi: http://doi.acm.org/10.1145/1163405.1163412

Mandal, A. K., Mandal, C., & Reade, C. M. P. (2006). Architecture of an automatic program evaluation system. CSIE. (http://sit.iitkgp.ernet.in/ chitta/pubs/CSIEAIT06-p152.pdf)

Mandal C., V. L. . R. C. M. P., Sinha. (2004). A web-based course management tool and web services. Electronic Journal of E-Learning, 2 . Retrieved from http://doi.acm.org/10.1145/1163405.1163412

Mansouri, F. Z., Gibbon, C. A., & Higgins, C. A. (1998). Pram: prolog automatic marker. In Iticse (p. 166-170).

Markiewicz, M. E., & de Lucena, C. J. P. (2001, July). Object oriented framework development. Crossroads, 7 , 3-9. (http://doi.acm.org/10.1145/372765.372771)

Mason, R., & Rehak, D. (2003). Keeping the learning in learning objects. In A. Littlejohn (Ed.), Reusing online resources: a sustainable approach to e-learning (pp. 20-34). London: Kogan Page. (http://oro.open.ac.uk/800/)

Massart, D. e. a. (2010). Taming the metadata beast: Ilox. D-Lib Magazine, 16 (11/12). (http://www.dlib.org/dlib/november10/massart/11massart.html)

McCallum, S. H. (2006). A look at new information retrieval protocols: Sru, opensearch/a9, cql, and xquery. In In world library and information congress (http://archive.ifla.org/IV/ifla72/papers/102-McCallum-en.pdf)

McGreal, R. (2008). A typology of learning object repositories. In H. H. Adelsberger, Kinshuk, J. M. Pawlowski, & D. G. Sampson (Eds.), Handbook on information technologies for education and training (p. 5-28). Springer Berlin Heidelberg.

Meier, W. (2002). exist: An open source native xml database. In Web-services, and database systems, node 2002 web and database-related workshops (pp.169-183). Springer.

Mory, E. H. (2007). Feedback Research Revisited. In D. H. Jonassen (Ed.), Handbook of research for educational communications and technology. Association for Educational Communications and Technology.

Nichani, M. (2001). Lcms = lms + cms [rlos] - how does this a ect the learner? the instructional designer? ELearningPost.

Nielsen, J. (1994). Usability engineering. San Francisco, Calif.: Morgan Kaufmann Publishers. Retrieved from http://www.worldcat.org/search?qt=worldcatorgallq = 0125184069

Nielson, F., Nielson, H. R., & Hankin, C. (1999). Principles of program analysis. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Ochoa, X., & Duval, E. (2009). Quantitative analysis of learning object repositories (Vol. 2) (No. 3). AACE. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5184802

Ochoa, X., Klerkx, J., Vandeputte, B., & Duval, E. (2011). On the use of learning object metadata: The globe experience. In Ec-tel (p. 271-284). (DBLP, http://dblp.uni-trier.de)

O'Kelly, J., & Gibson, J. P. (2006, June). Robocode & problem based learning: a non-prescriptive approach to teaching programming. SIGCSE Bull., 38 (3), 217-221. Retrieved from http://doi.acm.org/10.1145/1140123.1140182 doi: 10.1145/1140123.1140182

Oliveira, L., & Moreira, F. (2010). Personal learning environments: Integration of web 2.0 applications and content management systems. In 11th European conference on knowledge management (Vol. 2).

Oncu, C. H., S. (2011). Research in online learning environments: Priorities and methodologies. Computers and Education, 57 (1), 1098-1108. Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-78951474340partnerID=40md5=a3a5ec019895338110fc0a4cd4ed6dce
(cited By (since 1996) 3)

Pantel, C. (n.d.). A framework for comparing web-based learning environments (Unpublished master's thesis). School of Computing Science, Simon Fraser University, Canada.

Pisan, Y., Richards, D., Sloane, A., Koncek, H., & Mitchell, S. (2003). Submit! A web-based system for automatic program critiquing. In Proceedings of the australasian conference on computing education - volume 20 (pp. 59-68). Darlinghurst, Australia, Australia: Australian Computer Society, Inc. Retrieved from http://dl.acm.org/citation.cfm?id=858403.858411

Queirós, R. & Leal, J. P. (2012). Programming Exercises Evaluation Systems - An Interoperability Survey.. In M. Helfert, M. J. Martins & J. Cordeiro (eds.), CSEDU (1) (p./pp. 83-90), : SciTePress. ISBN: 978-989-8565-06-8

Queirós, R., & Leal, J. P. (2013). Making Programming Exercises Interoperable with PExIL. In J. Ramalho, A. Simões, & R. Queirós (Eds.) Innovations in XML Applications and Metadata Management: Advancing Technologies (pp. 38-56). Hershey, PA: Information Science Reference. doi:10.4018/978-1-4666-2669-0.ch003

Queirós, R. & Leal, J. P. (2012). PETCHA: a programming exercises teaching assistant. In Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education

(ITiCSE '12). ACM, New York, NY, USA, 192-197. DOI=10.1145/2325296.2325344 http://doi.acm.org/10.1145/2325296.2325344

Queirós, R. & Leal, J. P. (2013). Ensemble - An E-Learning Framework, Special issue on Cloud Education Environments at the Journal of Universal Computer Science (JUCS), DOI: 10.3217/jucs-018-11-1454

Reek, K. A. (1989, February). The try system -or- how to avoid testing student programs. SIGCSE Bull., 21 , 112-116. Retrieved from http://doi.acm.org/10.1145/65294.71198 doi: http://doi.acm.org/10.1145/65294.71198

Rehak, M. R., D. R. (2003). Keeping the learning in learning objects. In Littlejohn, A. (Ed.) Reusing online resources: a sustainable approach to e-Learning, 22-30.

Reilly, W., Wolfe, R., & Smith, M. (2006, April). Mit's cwspace project: packaging metadata for archiving educational content in dspace. Int. J. Digit. Libr., 6 (2), 139-147. Retrieved from http://dx.doi.org/10.1007/s00799-005-0131-2 doi: 10.1007/s00799-005-0131-2

Repository software survey. (2010). Repositories Support Project. Robertsson, E. (2002). Combining schematron with other xml schema languages (Tech. Rep.).

Rodriguez, E., Sicilia, M. A., & Arroyo, S. (2006). Bridging the semantic gap in standards-based learning object repositories. In Proceedings of the workshop on learning object repositories as digital libraries current challenges (pp. 478-483).

Rogers, S. A. (2003). Developing an institutional knowledge bank at ohio state university: From concept to action plan. portal Libraries and the Academy, 3 (1), 125-136. Retrieved from http://muse.jhu.edu/content/crossref/journals/portallibrariesandtheacademy=v003=3:1rogers:html

Romli, R., Sulaiman, S., & Zamli, K. (2010, june). Automatic programming assessment and test data generation a review on its approaches. In Information technology (itsim), 2010 international symposium in (Vol. 3, p. 1186 -1192). doi: 10.1109/ITSIM.2010.5561488

Saikkonen, R., Malmi, L., & Korhonen, A. (2001, June). Fully automatic assessment of programming exercises. SIGCSE Bull., 33 , 133-136. Retrieved from http://doi.acm.org/10.1145/507758.377666 doi: http://doi.acm.org/10.1145/507758.377666

Sampson, D. G., Zervas, P., & Chloros, G. (2012). Supporting the process of developing and managing lom application profiles: The ask-lom-ap tool. IEEE TRANSACTIONS ON LEARNING TECHNOLOGIES.

Schulte, C., & Bennedsen, J. (2006). What do teachers teach in introductory programming? In Proceedings of the second international workshop on computing education research (pp. 17-28). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/1151588.1151593 doi: 10.1145/1151588.1151593

Siddiqui, A., Khan, M., & Akhtar, S. (2008, August). Supply chain simulator: A scenario-based educational tool to enhance student learning. Comput. Educ., 51 (1), 252-261. Retrieved from http://dx.doi.org/10.1016/j.compedu.2007.05.008 doi: 10.1016/j.compedu.2007.05.008

Simon, B., Massart, D., van Assche, F., Ternier, S., Duval, E., Brantner, S., Mikl os, Z. (2005). A simple query interface for interoperable learning repositories. In Proceedings of the 1st workshop on interoperability of web-based educational systems (pp. 11-18).

Spacco, J., Hovemeyer, D., Pugh, W., Emad, F., Hollingsworth, J. K., & Padua-Perez, N. (2006, June). Experiences with marmoset: designing and using an advanced submission and testing system for programming courses. SIGCSE Bull., 38 (3), 13-17. Retrieved from http://doi.acm.org/10.1145/1140123.1140131 doi: 10.1145/1140123.1140131

Striewe, M., & Goedicke, M. (2010). Visualizing data structures in an e-learning system. In Csedu (p. 172-179).

Tang, Y. Y. T. . P. C. K., C. M. (2009a). An approach towards automatic testing of student programs using token patterns. In In proceedings of the 17th international conference on computers in education (icce 2009) (p. 188-190).

Tang, Y. Y. T. . P. C. K., C. M. (2009b). Automated systems for testing student programs: Practical issues and requirements. In In proceedings of the international workshop on strategies for practical integration of emerging and contemporary technologies in assessment and learning (p. 132-136).

Tang C.M., P. C., Yu Y. T. (2010). A review of the strategies for output correctness determination in automated assessment of student programs. In In proceedings of global chinese conference on computers in education.

Tastle, W. A., J., & Shackleton, P. (2005). E-learning in higher education: the challenge, e ort, and return of investment. International Journal on ELearning.

Team, J. (2006). E-learning repository systems research watch (Tech. Rep.). JISC.

Ternier, S. (2008). Standards Based Interoperability for Searching in and Publishing to Learning Object Repositories (Interoperabiliteit voor het publiceren en ontsluiten van leerobjecten in repositories met gebruik van standaarden) (Doctoral dissertation, K.U.Leuven). Retrieved from https://lirias.kuleuven.be/handle/123456789/242045

Ternier, S., Massart, D., Totschnig, M., Klerkx, J., & Duval, E. (2010). The simple publishing interface (spi). D-Lib Magazine, 16 (9/10). (http://www.dlib.org/dlib/september10/ternier/09ternier.html)

Trotteberg, H., & Aalberg, T. (2006). JExercise: A specification-based and test-driven exercise support plugin for Eclipse.

Tremblay, G., Gu erin, F., Pons, A., & Salah, A. (2008, March). Oto, a generic and extensible tool for marking programming assignments. Softw. Pract. Exper., 38 (3), 307-333. Retrieved from http://dx.doi.org/10.1002/spe.v38:3 doi: 10.1002/spe.v38:3

Truong, N. K. D. (2007). A web-based programming environment for novice pro- grammers (Doctoral dissertation, Queensland University of Technology). Retrieved from http://eprints.qut.edu.au/16471/

Tsunakawa, T. (2010). Pivotal approach for lexical translation (Unpublished doctoral dissertation). University of Tokyo.

Tzikopoulos, M. N. . V. R., A. (2009). An overview of learning object repositories. In In t. halpin (ed.), selected readings on database technologies and applications. IGI Global.

Varlamis, I., & Apostolakis, I. (2006). The present and future of standards for e-learning technologies. Interdisciplinary Journal of Knowledge and Learning Objects, 2 . (http://www.ijello.org/Volume2/v2p059-076Varlamis.pdf)

Verhoe, T. (2008). Programming task packages: Peach exchange format. International Journal Olympiads In Informatics, 2 , 192-207.

Wang, F. L., & Wong, T.-L. (2008). Designing programming exercises with computer assisted instruction. In Proceedings of the 1st international conference on hybrid learning and education (pp. 283-293). Berlin, Heidelberg: Springer-Verlag. Retrieved from http://dx.doi.org/10.1007/978-3-540-85170-725 doi: 10.1007/978-3-540-85170-725

Ward, J. (2004). Unqualified dublin core usage in oai-pmh data providers. OCLC Systems & Services, 20 (1), 40-47. (http://dx.doi.org/10.1108/10650750410527322)

Wiedenbeck, S., Labelle, D., & Kain, V. N. R. (2004). Factors affecting course outcomes in introductory programming. In In 16th annual workshop of the psychology of programming interest group (pp. 97-109).

Williams, G. M., J. (2005). The evolution of e-learning. Universitas 21 Global.

Wilson, S., Blinco, K., & Rehak, D. (2004, July). Service-Oriented Frameworks: Modelling the infrastructure for the next generation of e-Learning Systems (Tech. Rep.). JISC Report. (http://www.jisc.ac.uk/uploadeddocuments=AltilabServiceOrientedF rameworks:pdf)

Xavier, J., & Coelho, A. (2011, 14-16 November, 2011). Computer-based assessment system for e-learning applied to programming education. In Iceri2011 proceedings (p. 3738-3747). IATED.

KEY TERMS & DEFINITIONS

Assessment Systems – systems responsible for the evaluation of computer programs based on several evaluation models such as test cases.

Learning Tools Interoperability - is a specification developed by IMS Global Learning Consortium. The principal concept of LTI is to establish a standard way of integrating rich learning applications (often remotely hosted and provided through third-party services) with platforms like learning management systems, portals, or other educational environments. In LTI these learning applications are called Tools (delivered by Tool Providers) and the LMS, or platforms, are called Tool Consumers.

learning object repository- is a type of a digital library. It enables educators to share, manage and use educational resources. A more narrow definition would also require that repositories implement a metadata standard

Massive Open Online Courses: A massive open online course (MOOC) is an online course aimed at unlimited participation and open access through the Web. In addition to traditional course materials such as videos, readings, and problem assets, MOOCs provide interactive user forums that help build a community for students, teachers and teaching assistants (TAs).