Ensemble – an innovative approach to practice computer programming

Ricardo Queirós
José Paulo Leal
CRACS/INESC TEC & ESEIG/IPP, Polytecnic Institute of Porto, Portugal
CRACS/INESC TEC & Department of Computer Science, Faculty of Sciences, University of Porto, Portugal

ABSTRACT

Currently the teaching-learning process in domains, such as computer programming, is characterized by an extensive curricula and a high enrolment of students. This poses a great workload for faculty and teaching assistants responsible for the creation, delivering and assessment of student exercises.
The main goal of this work is to foster practice-based learning in complex domains. This objective is attained with an e-learning framework - called Ensemble - as a conceptual tool to organize and facilitate technical interoperability among services. The Ensemble framework is used on a specific domain - computer programming. Content issues are tacked with a standard format to describe programming exercises as learning objects. Communication is achieved with the extension of existing specifications for the interoperation with several systems typically found in an e-learning environment.
In order to evaluate the acceptability of the proposed solution an Ensemble instance was validated on a classroom experiment with encouraging results.

Key words: Learning Objects, Assessment, Standards, Interoperability, programming languages, repositories.

INTRODUCTION

For someone to acquire, improve or even maintain a complex skill it is necessary to practice it on a regular basis (Gross, 2005), (Eckerdal, 2009). The amount of practice required depends on the nature of the activity and on each individual. How well an individual improves with practice is directly related with its inherent capabilities, its previous know-how about the domain and the type of feedback that is available for improvement. If feedback is either non-existent or inappropriate, then the practice tends to be ineffective or even detrimental to learning.

An apt example of a complex skill is music. Learning music requires discipline and perseverance while acquiring the concept of reading scores, practising an instrument or playing with a group. In order to enhance these skills and motivate young students, instructors use e-learning, mainly in introductory courses, to make the learning of music more appealing. One good example is the NoteFlight[1] web application (Figure 1), a tool designed to teach music by creating, viewing, printing and hearing music notation. The tool was recently integrated[2] with Moodle, a popular LMS. This integration enables instructors to create assignments (e.g. giving students a partial composition to be completed), to manually grade the student submissions and to give them feedback promptly.

[1] NoteFlight web site: http://www.noteflight.com
[2] NoteFlight demo with Moodle integration: http://videos.noteflight.com/MoodleBasicLTI.mov

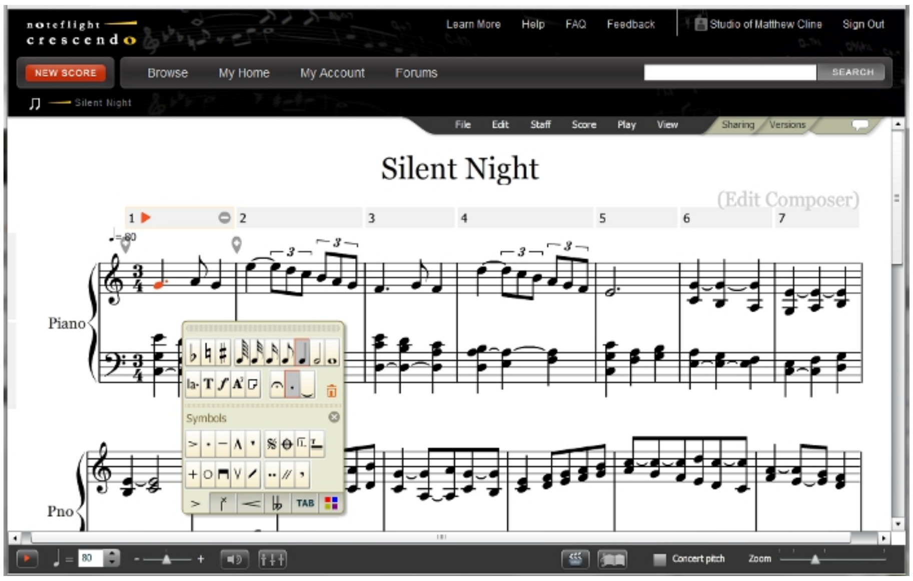

***Figure 1 –** Noteflight application.*

Besides music, there are other areas where evaluation is a key component in practice such as management, health sciences, electronics. 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 learning processes that require the use of special authoring, rendering and assessment tools. These tools should be integrated in instructional environments in order to provide a better learning experience. However, these tools would be too specific to incorporate in an e-learning platform. Even if they could be provided as pluggable components, the burden of maintaining them would be prohibitive to institutions with few courses in those domains.

The motivation for this work comes from yet another domain with complex evaluation: computer programming. Introductory programming courses are generally regarded as difficult and often have high failure and dropout rates (Ala-Mutka, 2005), (O'Kelly, 2006), (Robins, 2003). Researchers pointed out several causes for these rates (Esteves, 2010). The most consensual are: teaching methods based on lectures about programming language syntaxes, subject complexity (learning how to program means to integrate knowledge of a wide variety of conceptual domains such as computer science and mathematics while developing expertise in problem understanding, problem-solving, unit testing and others) and student's motivation (the public image of a "programmer" as a socially inadequate "nerd" and the reputation of programming courses as being extremely difficult affects negatively the motivation of the students).

Many educators claim that "learning through practice" is by far the best way to learn computer programming and to engage novice students (Gross, 2005), (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. An exercise solved wrong will consolidate a false belief, and without feedback many students will not be able to overcome their difficulties.

A number of learning tools and environments have been built to assist both teachers and students in introductory programming courses.

Rongas, Kaarna, and Kalviainen (Rongas, 2004) established a classification for these tools dividing them into four categories: 1) integrated development interfaces, 2) visualization tools, 3) virtual learning environments, and 4) systems for submitting, managing, and testing exercises. To the best of the author's knowledge, no e-learning environment described in the literature integrates all these facets (Verdu, 2011), (Gomes, 2007), (Esteves, 2010).
For instance, an evaluator should be able to participate in learning scenarios where teachers can create exercises, store them in a repository and reference them in a LMS and students solve the exercises in integrated development environments and submit to evaluators who delivers an evaluation report back to students.
Several systems (Somyajit, 2008), (Verdu, 2011), (Xavier, 2011), (Guerreiro, 2008) try to address this issue allowing the integration of automatic assessment tools with course management systems but these approaches rely on ad hoc solutions or proprietary plug-ins rather on widely accepted international specifications for content description and communication among systems.

In this paper we present a proposal for an e-learning framework called the Ensemble E-Learning Framework (EeF). The EeF is a conceptual tool to organize a network of e-learning systems and services based on content and communications specifications. The EeF is the basis for the design and implementation of Ensemble instances as realizations of the framework. In this work we present the specialization of the Ensemble framework for a specific domain: the computer programming learning domain. In order to validate the instance, we use the Ensemble instance in a Polytechnic School and we present the main results.

The structure of the chapter is as follows. The second section presents a brief background on the topics of e-learning standards and systems. The third and fourth section details the Ensemble framework specialization for the computer programming domain. The next section validates the use of an Ensemble instance in classroom. We conclude the chapter with the presentation of the main contributes of this work and a brief discussion of future research.

BACKGROUND

The Ensemble framework recommends the use of several types of e-learning systems, as well of several international e-learning specifications. Thus, a detailed study was made to identify the most prominent and mature e-learning systems and specifications.

E-Learning Systems

The evolution of eLearning can be summarized by the transition of the early monolithic systems developed for specific learning domains to the new systems that can invoke specialized services and be plugged in any eLearning environments. These types of systems evolved from Content Management Systems (CMS). The CMS was introduced in the mid-1990s mostly by the online publishing industry. This type of system can be defined as a data repository that also includes tools for authoring, aggregating and sequencing content in order to simplify the creation, administration and access to online content (Nichani, 2009). The content is organized in small self-contained pieces of information to improve reusability at the content component level. These content components when used in the learning domain are called "learning objects" (LO) and the systems that manage them are called Learning Content Management Systems (LCMS) (Leal & Queirós, 2010). LO are context independent, transportable and reusable pieces of instruction that are digitally managed and delivered (Rehac, 2003). There are other definitions for Learning Objects (LO). Rehak & Mason (Rehac, 2003) define a learning object as: "a digitized entity which can be used, reused or referenced during technology supported learning".

Nowadays services play a crucial role in learning processes. These services can be easily recombined in different learning processes to assist on the teaching-learning process. This chapter focuses on learning processes within domains with complex evaluation such as computer programming learning. In this domain there are several candidates to offer services such as those referred by Rongas, Kaarna, and Kalviainen (Rongas, 2004), namely: LMSs, LORs and ASs.
In the LMS side, an interoperability comparative study (Leal, 2012) was conducted to select an LMS on which to base the development of e-learning systems integrating heterogeneous components. Two LMSs vendors were chosen (Moodle and Blackboard) and their interoperability features were analysed by splitting in two facets (learning content management and academic management) reflecting the broad classes of systems of a typical LMS operational environment.
In the LOR side, an interoperability study (Queirós, 2013) was conducted on the existing repository software distinguishing two categories of repositories - digital libraries and learning objects repositories. It also focused on the distinction between the actual repositories and the software used to implement them, highlighting the differences in their interoperability features.
Finally, a survey (Queirós, 2012) was made on ASs for programming exercises focusing on their interoperability features. The survey gathered information on the interoperability features of the existent ASs and compared them regarding a set of predefined criteria such as content specification and interoperation standards with other tools.

E-Learning Standards

In the eLearning context, standards are generally developed for the purposes of ensuring interoperability and reusability in systems and of the content and meta-data they manage. In this context, several organizations (IEEE, AICC, IMS, ADL) have been developed standards and specifications regarding the creation of standards, specifications, guidelines, best practices on the description and use of eLearning content. In the last decade practitioners of eLearning started valuing more the interchange of course content and learners' information, which led to the definition of new standards. These standards are generally developed to ensure interoperability and reusability in content and communication. In this context, several organizations (IMS, IEEE, ADL, ISO/IEC) have developed specifications and standards (Friesen, 2005). These specifications define, among many others, standards related to learning objects (Rehak and Mason, 2003), such as packaging them, describing their content, organizing them in modules and courses and communicating with other eLearning systems.
Packaging is crucial to store eLearning material and reuse it in different systems. The most widely used content packaging format is the IMS Content Packaging. An IMS CP learning object assembles resources and meta-data into a distribution medium, typically an archive in zip format, with its content described in a manifest file in the root level. The manifest file - named imsmanifest.xml - adheres to the IMS CP schema and contains several sections such as Metadata to describe the package as a whole and Resources to refer to resources (files) needed for the manifest and metadata describing these resources.
The metadata included in the manifest uses another standard - the IEEE Learning Object Metadata. The IEEE LOM is a data model used to describe a learning object. The model is organized in several categories that cover general data, such as title and description, technical data such as object sizes, types and durations, educational characteristics and intellectual property rights, among many others. These categories are very comprehensive and cover many facets of a LO. However, LOM was designed for general LO and does not to meet the requirements of specialized domains, such as the automatic evaluation of programming exercises. For instance, there is no way to assert the role of specific resources, such as test cases or solutions. Fortunately, IMS CP was designed to be straightforward to extend through the creation of application profiles.
The term Application Profile generally refers to "the adaptation, constraint, and/or augmentation of a metadata scheme to suit the needs of a particular community". A well know eLearning application profile

is SCORM that extends IMS CP with more sophisticated sequencing and Contents-to-LMS communication.
The IMS GLC is also responsible for another application profile, the Question & Test Interoperability specification. QTI describes a data model for questions and test data and, since version 2.0, extends the LOM with its own meta-data vocabulary. QTI was designed for questions with a set of pre-defined answers, such as multiple choice, multiple response, fill-in-the-blanks and short text questions. Although long text answers could be used to write the program's source code, there is no way to specify how it should be compiled and executed, which test data should be used and how it should be graded. For these reasons we consider that QTI is not adequate for automatic evaluation of programming exercises, although it may be supported for sake of compatibility with some LMS. Recently, IMS Global Learning Consortium proposed the IMS Common Cartridge that adds support for several standards (e.g. IEEE LOM, IMS CP, IMS QTI, IMS Authorization Web Service) and its main goal is to shape the future regarding the organization and distribution of digital learning content.
The standardization of the learning content it is not enough to ensure interoperability, which is a major user concern with the existing systems (Leal and Queirós, 2010). In the last few years there have been initiatives (Holden, 2004) to adapt Service Oriented Architectures (SOA) to eLearning. These initiatives, commonly named eLearning frameworks, had the same goal: to provide flexible learning environments for learners worldwide. While eLearning frameworks are general approaches for eLearning system integration, several organizations proposed service oriented approaches specifically targeted to the LMS (IMS Digital Repository Interoperability – IMS DRI) and Repositories (IMS LTI).
The IMS DRI specification provides a functional architecture and a reference model for repository interoperability composed by a set of recommendations for common repository functions, namely the submission, search and download of LOs. It recommends the use of web services to expose the repository functions based on the Simple Object Access Protocol (SOAP) protocol, defined by W3C. Despite the SOAP recommendation, other web service interfaces could be used, such as, Representational State Transfer (REST) (Fielding, 2005).
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 LMSs. The main goal of the LTI is to standardize the process for building links between learning tools and the LMS. The IMS launched also a subset of the full LTI v1.0 specification called IMS Basic LTI. This subset exposes a unidirectional link between the LMS and the application. For instance, there is no provision for accessing run-time services in the LMS and only one security policy is supported.

THE ENSEMBLE FRAMEWORK INSTANCE

This chapter presents a specialization of the Ensemble framework (Queirós & Leal, 2013) for a new domain - computer programming. The presentation includes the overall architecture of the Ensemble instance followed by the description of the data and the integration models. The data model relies on the PExIL specification - an interoperability language for describing programming exercises - included as a LAO resource in an IMS CC package as recommended by the Ensemble specification. The integration model detailed how systems and services of this Ensemble instance are connected through the extension of the recommended specifications of the EeF. The integration model includes the creation of a new service for the communication with the AS based on the Evaluate service genre and the extension of the IMS DRI and LTI specifications for the integration of LOR and LMS, respectively. Then, a workflow of a network based on this Ensemble instance was presented showing how systems and services interact. Finally, a selection of tools adjusted to the models was presented. This process was straightforward evidencing the interoperability features of the framework. Three of these systems and services that integrate this network were created from scratch, more precisely, the crimsonHex repository, the BabeLO converter and the Petcha Teaching Assistant. The acceptability of this Ensemble instance is validated in next chapter through an experiment in a pedagogical environment.

Architecture

This section presents the overall architecture of a network of e-learning systems and services participating in the automatic evaluation of programming exercises. The architecture (Figure 2) is composed by the following components:

- Teaching Assistant (TA) to interface the users with the network and to mediate the communication among all components;
- Integrated Development Environment (IDE) to code the exercises;
- Assessment System (AS) to evaluate exercises of students;
- Learning Objects Repository (LOR) to store and retrieve exercises;
- Learning Management System (LMS) to present the exercises to students;
- Conversion System (CS) to convert exercises formats.

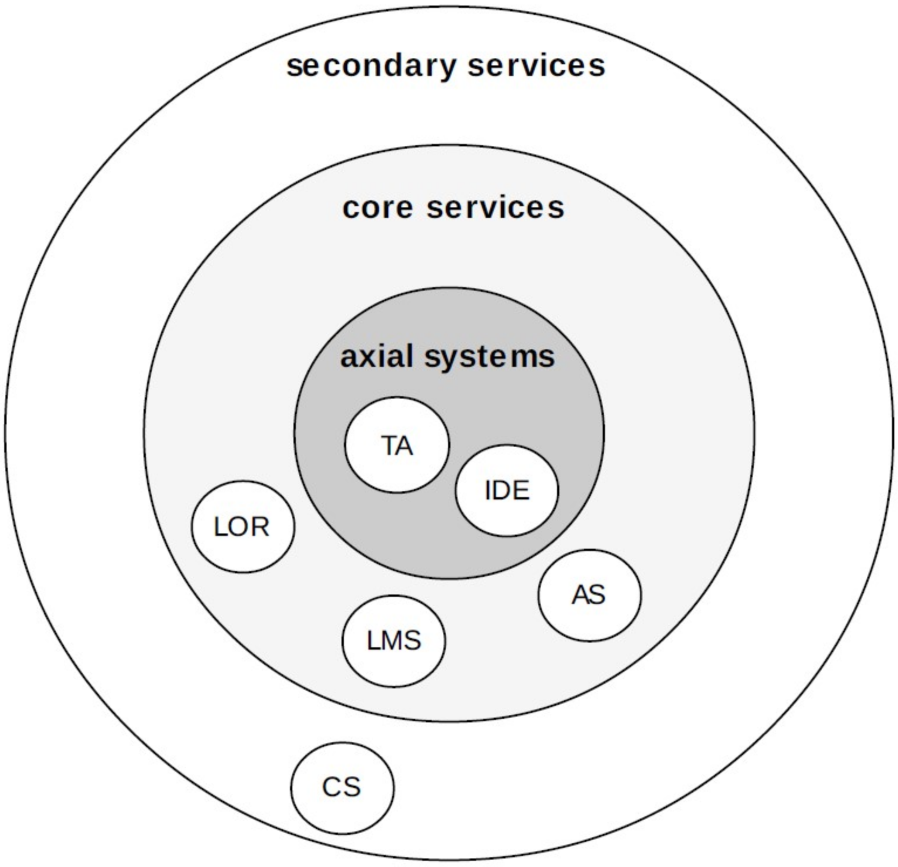

Figure 2 *–Overall architecture of the Ensemble instance*

Data Model

The concept of Learning Object (LO) is fundamental for producing, sharing and reusing content in e-Learning (Friesen, 2005). In essence a LO is a container with educational material and metadata describing it. Since most LOs just present content to students they contain documents in presentation formats such as HTML and PDF, and metadata describing these documents using LOM (or other metadata format). When a LO includes exercises to be automatically evaluated by an e-learning system, it must contain a document with a formal description for each exercise. The QTI specification is an example of a standard for this kind of definitions that is supported by several e-learning systems. However, QTI was designed for questions with predefined answers and cannot be used for complex evaluation domains such as the programming exercise evaluation (Leal & Queirós, 2009). In particular, the programming exercise domain requires interdependent resources (e.g. test cases, solution programs, exercise description) usually processed by different services in the programming exercise life-cycle. This kind of data cannot be characterized as metadata as they are data effectively needed for evaluation.

The data model of the Ensemble instance extends the IMS CC specification for describing programming exercises. This extension is achieved by adding a new LAO resource in the CC package (Figure 3) as recommended by the Ensemble specification.

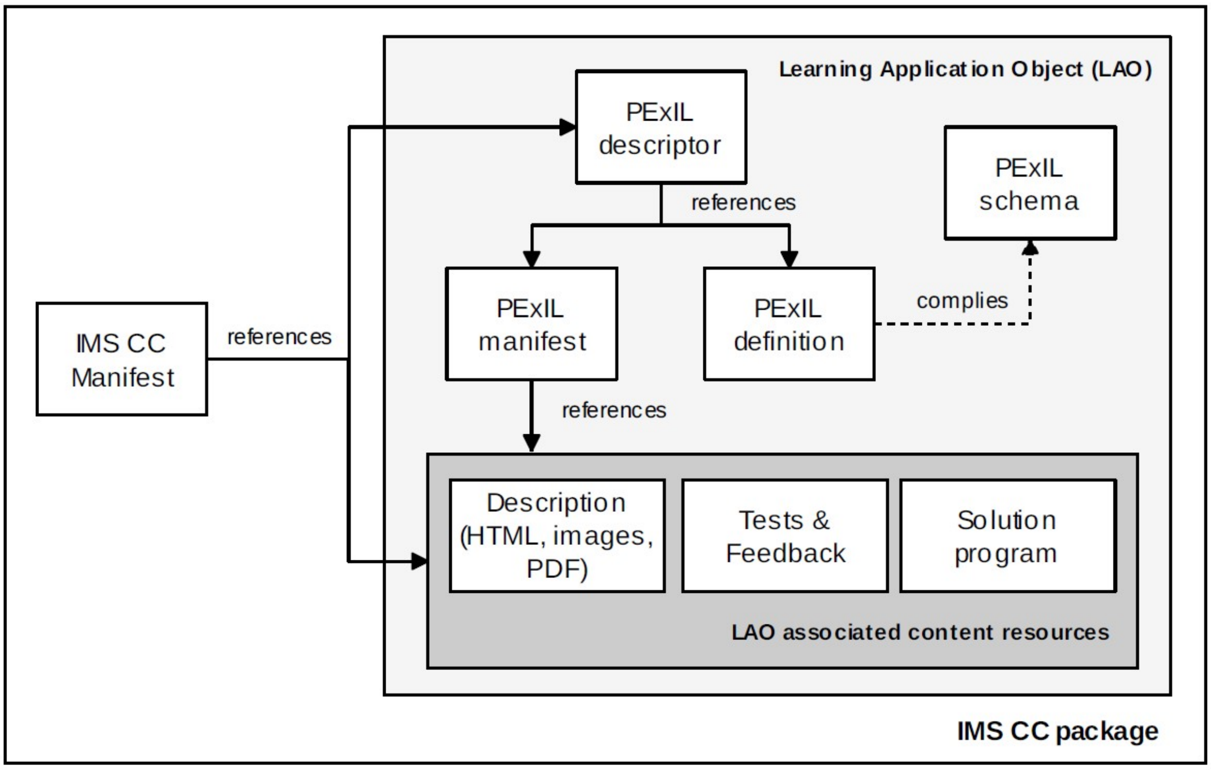

Figure 3 *– Programming exercise as a IMS CC package*

A LAO resource contains an XML descriptor that serves as the entry point for the target system. The descriptor file is not intended to be displayed within the target system. Rather, it is intended to be processed by the target system upon import of the cartridge. The descriptor file (pexil.xml) is associated with a LAO by means of a file element. It includes references for two XML documents: pexildefinition.xml and pexilmanifest.xml. The former describes all data needed for the generation of the evaluation resources. The latter is a manifest with references for all the evaluation resources generated. These resources - called associated content resources - comprise the exercise description, tests and feedback files and the solution program.

Integration model

The integration model depicted in Figure 4 relies on communication standards recommended by the Ensemble specification. This network model abstracts specific systems and focuses on system types. For instance, it is possible to use in 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.

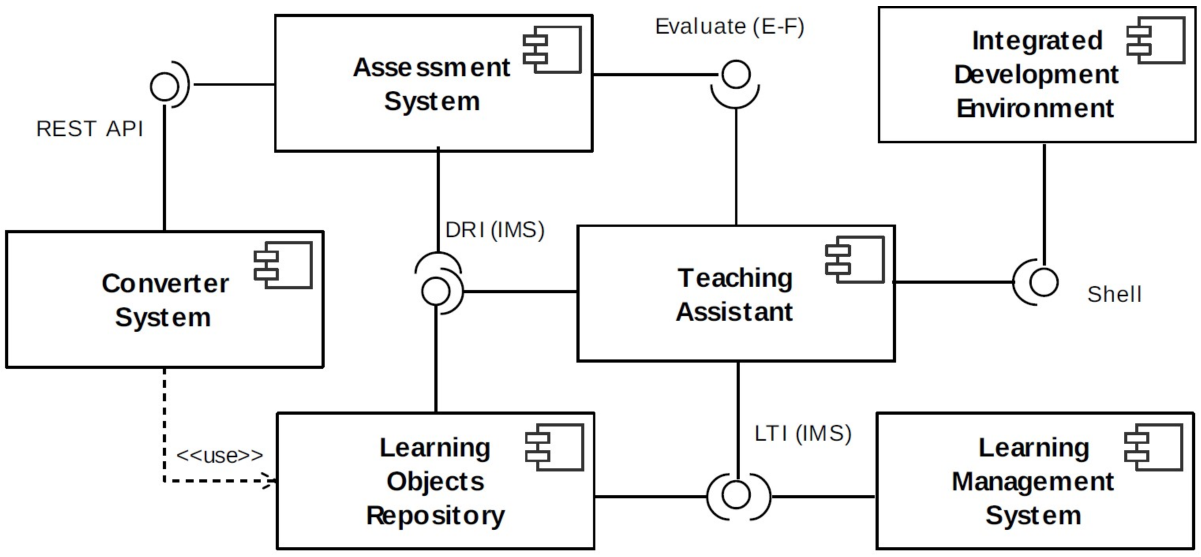

Figure 4 – *Network component diagram*

The previous subsections report on the efforts made to integrate the systems and services of an Ensemble instance for the computer programming domain. These efforts included the creation and extension of communication specifications such as:

- the extension of the IMS DRI specification for the integration of LOR;
- the extension of the IMS LTI specification for the integration of LMS;
- the creation of an evaluation service for the communication with the AS.

For each specification the interface definition and response bindings were presented. This subsection summarizes all the interactions among systems and services based on these specifications and represented in the UML sequence diagram depicted in Figure 5.

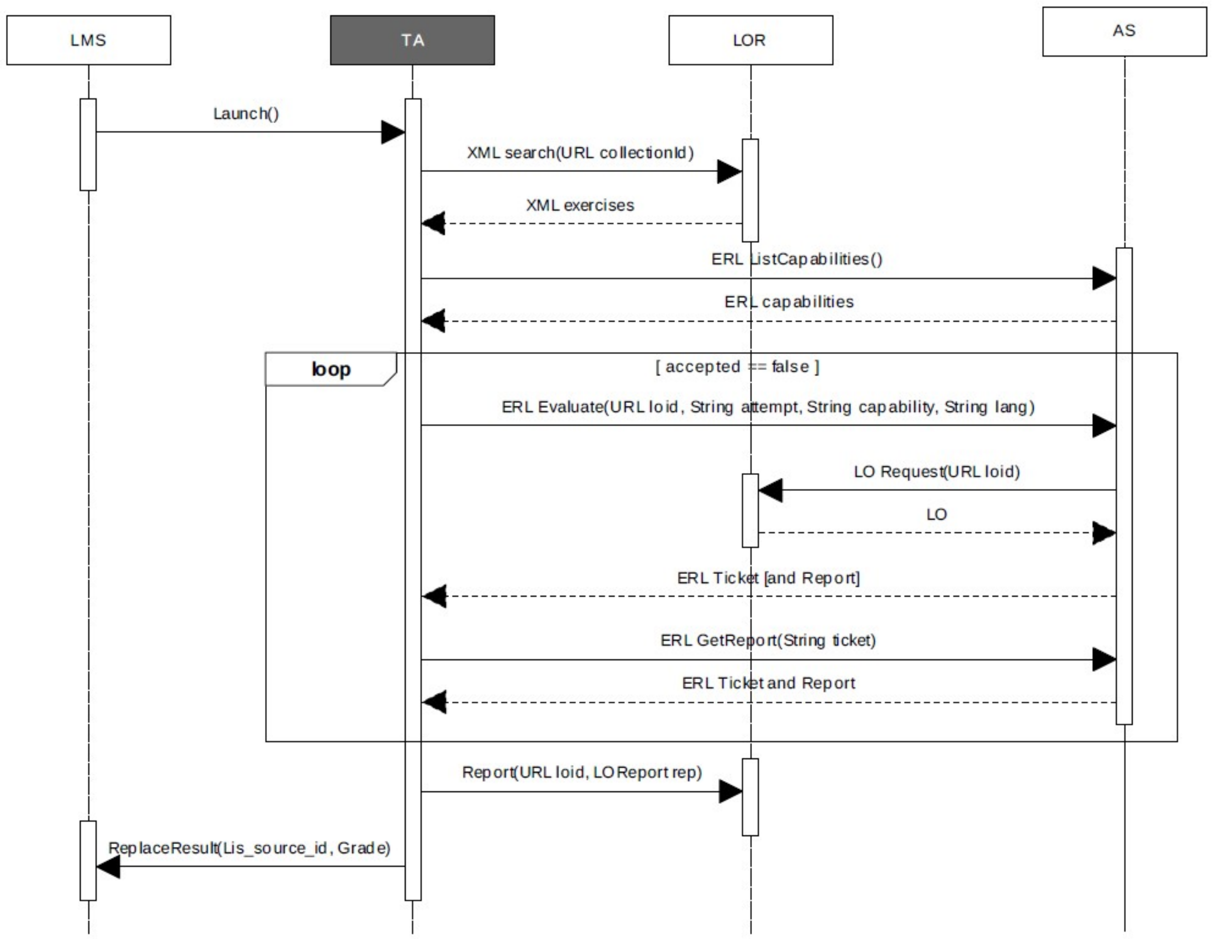
LMS
TA
LOR
AS
Launch()
XML search(URL collectionId)
XML exercises
ERL ListCapabilities()
ERL capabilities
loop
[accepted == false]
ERL Evaluate(URL loid, String attempt, String capability, String lang)
LO Request(URL loid)
LO
ERL Ticket [and Report]
ERL GetReport(String ticket)
ERL Ticket and Report
Report(URL loid, LOReport rep)
ReplaceResult(Lis_source_id, Grade)

Figure 5 *–Sequence diagram of the Ensemble instance.*

The workflow presented in Figure 5 starts by the selection of an LTI activity by the student. This activity was previously configured by the teacher by selecting a collection of exercises. After selection, the LMS launches the TA through the Launch function of the LTI specification passing the user and course parameters. In order to present the programming exercises to the student the TA forwards the search to the LOR using the Search DRI function with the given collection URL. Meanwhile the TA gets from the AS the evaluation capabilities for later evaluation using the ListCapabilities function. In general each student is able to make several submissions for the same exercise and an activity may include several exercises. Each evaluation starts with an Evaluate request from the TA to the AS, sending a program and referring an exercise and a programming language. The AS retrieves the LO from the repository to have access to test cases, special correctors and other metadata. The AS responds with a ticket and an evaluation report, if the evaluation is completed within a certain time frame. The TA may retrieve the evaluation report using the GetReport function with the ticket as argument. When the student ends the session, the TA sends an usage report to the LOR using the DRI report function and a grade back to the

LMS using the LTI replaceResult function. This latter feature is not yet implemented due to the missing Full LTI support by the most important LMS.

Tools Selection

The data and integration model of this Ensemble instance relies on content and communication standards as recommended by the Ensemble specification. These interoperability effort abstracts specific systems and focus on system types. This approach has facilitated the selection of tools for deployment purposes. The next paragraphs discuss the selection of each type of system or service.

On the LMS side the choice fell on Moodle since it is a popular and open source LMS, arguably the most popular LMS nowadays (Davis, 2009), (Cole, 2007). 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 - 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. This was the logical choice after the survey made to 15 assessment systems. One of the most important reasons for its selection was the support of web services.

The IDE system selected was Visual Studio Express for C\# assignments. Successful tests were made also with the Eclipse IDE for JAVA assignments on this network.

The TA system selected was Petcha (Queirós & Leal, 2012). Petcha has a two-fold goal: to coordinate the systems and services of this network and to interface with users, both teachers and students. Given the specificity of this role no other similar systems were found.

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.

ENSEMBLE EVALUATION

This chapter evaluates the Ensemble instance. The Nielsen's model (Nielsen, 1994) is used to evaluate the acceptability of Petcha as the user interface of the network. For this evaluation an experiment was conducted with undergraduate students and their teachers. This chapter starts by presenting the evaluation model followed in the experiment based on the heuristics of Nielsen. Then, the design of the experiment is described. The description includes the methodology followed, the educational context and infrastructure where the experiment occurs and the instruments used to collect the data. Finally, the evaluation results are presented and analysed.

Evaluation Model

According to Nielsen (Nielsen, 1994) the acceptability of a system is defined as the combination of social and practical acceptability. The former determines the success/failure of the system, since the more the system is socially acceptable the greater is the number of people using it. The latter relates factors such as usefulness, cost, reliability and interoperability with existing systems. An adaptation to the Nielsen's model (Nielsen, 1994) is depicted in Figure 6.

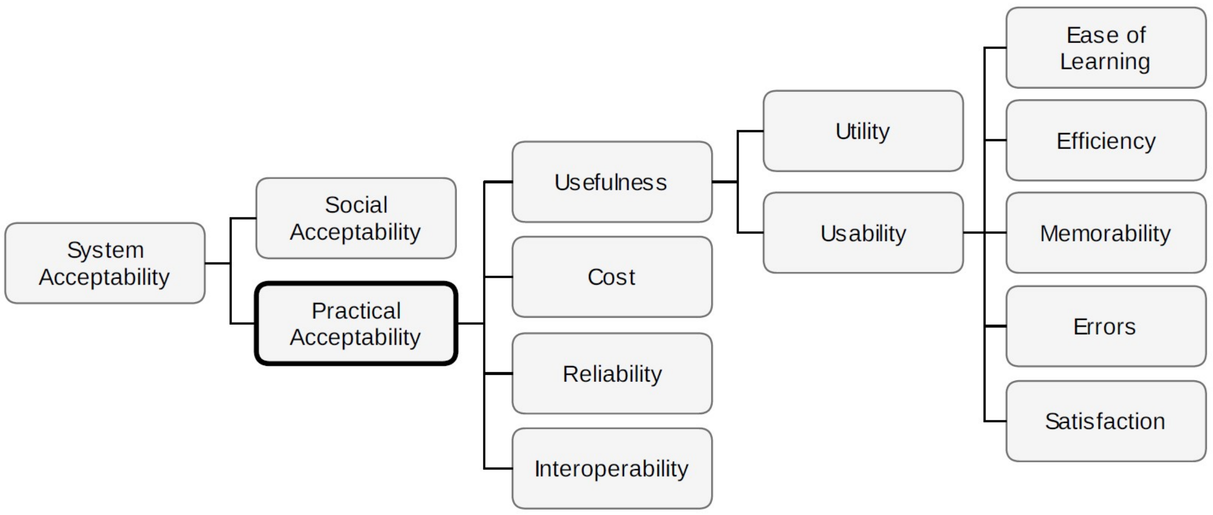

Figure 6 *– System acceptability. Adapted from Nielsen.*

The **usefulness** factor relates the utility and usability offered by the system. Utility is the capacity of the system to achieve a desired goal. As the system perform more tasks, more utility he has. Usability is defined by Nielsen as a qualitative attribute that estimates how easy is to use an user interface. He mentions five characteristics involved in the concept of usability: **ease of learning** - the system should be easy to learn so that the user can start doing some work with the system; **efficiency** - the system should be efficient to use, so after the user learns the system, a high level of productivity is possible; **memorability** - the system should be easy to remember so that the casual user is able to return to the system after a period without using it, without requiring to learn it all over again; **errors** - the system should prevent the user from committing errors as should deal with them gracefully and minimizing the chance of occurring catastrophic errors; **satisfaction** - the system should be pleasant to use so that users become subjectively satisfied when using. The **cost** factor was not considered since Petcha is a free (and open source) software. The **reliability** is the ability of a system or component to perform its required functions under stated conditions for a specified period of time. **Interoperability** is the ability of two or more systems or components to exchange information and to use the information that has been exchanged.

This chapter presents an acceptability evaluation on an Ensemble network for the computer programming domain. To carry out this evaluation an experiment was conducted in a higher education school. This evaluation focuses on three facets: usefulness, reliability and interoperability.

For the usefulness facet, a questionnaire based on the Nielsen's heuristics was filled in by students in the end of that experiment. On one hand, the results showed that the aesthetic was the heuristic with higher values of satisfaction. The respondents claim that the minimalist design is one of the strongest points of Petcha (the front-end of the Ensemble network). On the other hand, the results reveal deficiencies in three areas: flexibility, freedom and documentation. In the flexibility heuristic, the respondents felt difficulties in personalize the user interface and speed up the execution of frequent actions (e.g. definition of short-cuts). The freedom heuristic was another area where students were not satisfied since Petcha does not allow, for instance, the undo/redo of actions. Finally, students complained of the unavailability of supporting documentation while using Petcha.
In the reliability facet students stated difficulties in understanding error messages and teachers complained about the lack of intuitiveness in the creation of exercises.
For the interoperability facet, the communication between pairs of systems was analysed during the experiment by comparing the expected values with the real values (those obtained through service logs). The results show that the proposed e-learning framework is useful in practice. The figures collected during the experiment are within reasonable bound from the expected values. These results show that the network is stable enough to handle a significant number of students, and exercises and can be used in a classroom setting.

Experiment Design

The experiment took place at the Escola Superior de Estudos Industriais e de Gestão (ESEIG) - a school of the Polytechnic Institute of Porto - during the months of October and November of 2011. The participants were students from the first-year of the course Algorithms and Programming and their teachers. This course is offered to the degree in Mechanical Engineering and aims to introduce students to programming concepts.

The experiment methodology followed the experimental research method (Oncu, 2011). For this purpose experimental and control groups (two classes from the same course) were settled. The first class (the experimental group) had 21 students and the second class (the control group) had 19 students.
Students of both groups have similar characteristics such as the gender and previous background. For instance, the gender in both groups were well distributed, respectively 62% and 57% of females in both groups.
The conditions of the experiment were also equal for both classes (e.g. syllabus, teaching times, teacher, labs, technical means).

Although it could be assumed that the population of the classes were almost randomly formed, strictly the experiment should be called a **quasi-experiment**. A completely randomised design was not possible due to operational reasons. Based on this type of design, a **static group comparison design** (Borg, 1971) was followed where students of class A used Ensemble (the experimental group) and students of class B did not use it (the control group).

The course has an average enrolment of 40 students per year divided in two classes. The course is organized in two lectures of one hour each and one lab session of 4 hours per week. The experiment occurred in 6 lab sessions. In each lab session both groups (a total of 40 students) had 3 exercises to solve. In the experimental group the teacher only intervenes to solve operational issues related to the use of Ensemble and does not give any feedback to students regarding the exercises. Prior to the experiment, teacher and students were prepared for the experiment.

The instruments used for collecting data on the experiment were the following: surveys (session & final survey), service logs, students' attendance logs and grades.

The **surveys** were fulfilled and collected on-line using Google Forms[3]. Two types of surveys were presented to students: session and final survey (Appendix 1). The former was filled in by both groups of students after each lab session. The questionnaire[4] includes questions on the number of solved exercises and the feedback impact. It had an average of 38 responses per session (the equivalent to 95% of the total of students). The latter was presented to the experimental group at the end of the experiment. The questionnaire includes questions on the Petcha usefulness and reliability. The final survey was completed by all the students from the experimental group.

The **service logs** were used to attest the accuracy of the experimental group questionnaires responses. The data collected in the surveys of the experimental class was checked against the logs of Petcha and other systems in the network. An average discrepancy of 4.6% between these two sets of values was found.

The **student attendance** and **student outcomes** (programming module and semester grades) were collected through the Academic Management System used at ESEIG. The data was exported to a spreadsheet to simplify the data processing.

[3] http://www.google.com/google-d-s/forms/
[4] http://goo.gl/AlhsL

Results and discussion

In this section the Ensemble network is evaluated based on three Nielsen heuristics: usefulness, reliability and interoperability.

Usefulness

Usefulness combines the utility and usability of a system. Utility is the capacity of the system to achieve a desired goal. Usability is defined by Nielsen as a qualitative attribute that estimates how easy is to use an user interface.

Petcha was evaluated according to the Nielsen's model using a heuristic evaluation methodology. A heuristic evaluation is an inspection method for computer software that helps to identify usability problems in the user interface (UI) design. Jakob Nielsen's heuristics (Appendix 2) are arguably the most used usability heuristics for user interface design. Based on these heuristics a questionnaire (with 41 questions was designed in Google Forms. The aim of the questionnaire is to identify usability problems in the UI design of Petcha. Two profiles of users answered this questionnaire in the end of the experiment: 21 students (experimental group) and 4 teachers (although only one was the teacher of the students that participated in the experiment).

Figure 7 shows the results obtained grouped by the Nielsen's heuristics. The data collected are shown in graphs. In the chart graphs the heuristics are sorted in ascending order of user satisfaction.

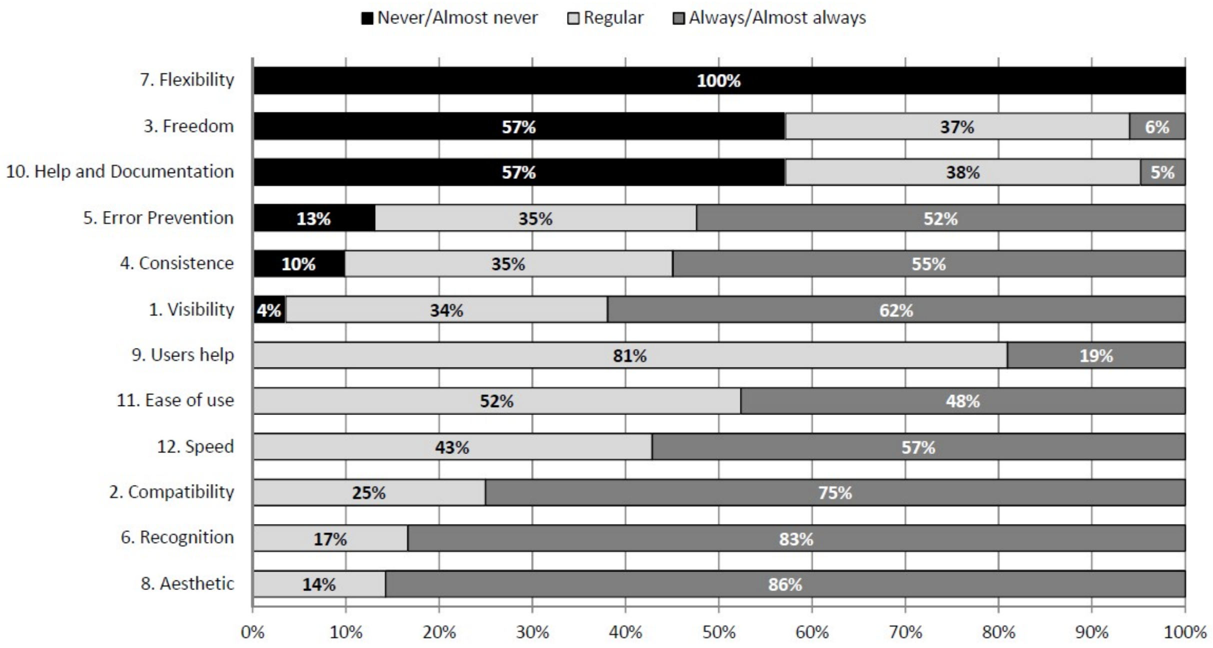

***Figure** 7 –Results of each heuristic in the student's profile.*

The results highlight deficiencies in three areas: Flexibility, User control and freedom, Help and documentation.
In regard to the flexibility of the system, respondents considered that the system do not allow the personalization of the interface as shown in Figure 8, more precisely, the activation/deactivation of certain functions and the configuration of the screens. The possibility of use of accelerator keys to speed up the interaction with the TA is also a handicap of Petcha.

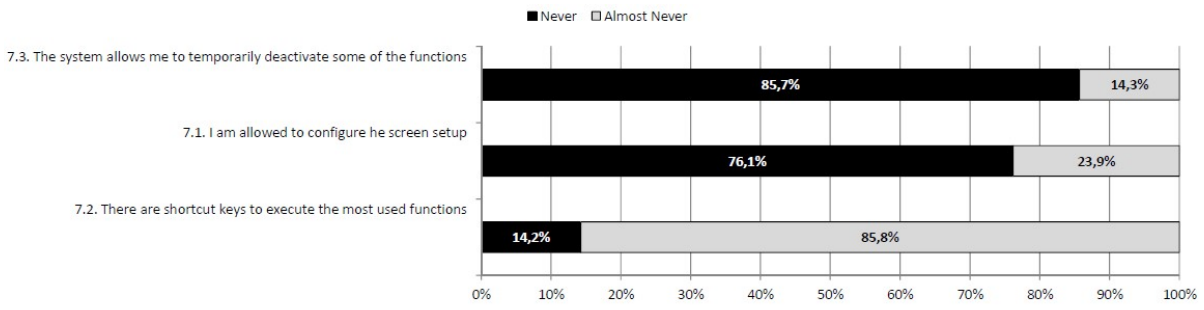

***Figure 8** – Evaluation of Petcha's flexibility.*

The User control and freedom is the second worst facet (Figure 9). Most of the complaints focused on the inability to cancel or roll back mistakes made to a previous and safe state.

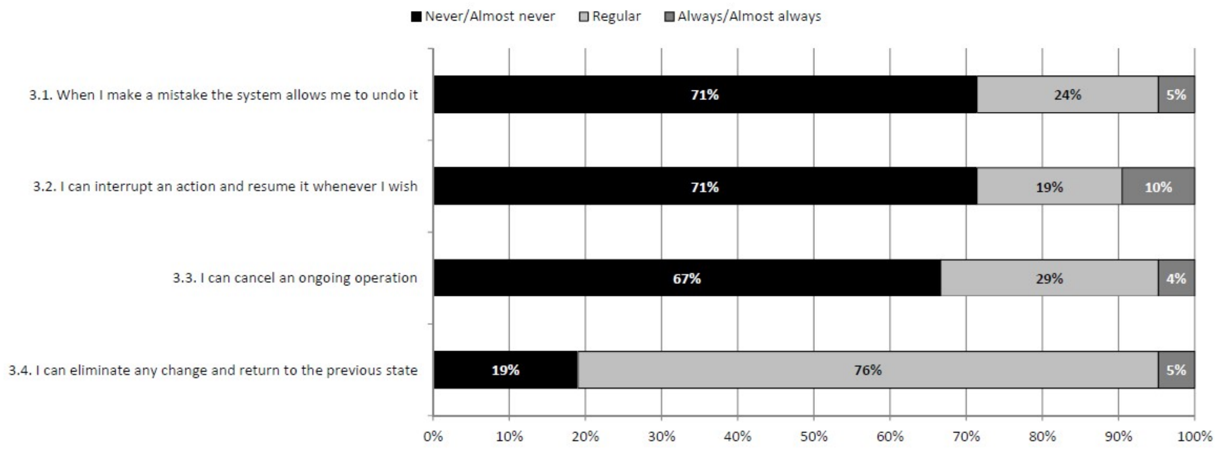

***Figure 9** – Evaluation of Petcha's freedom.*

The Help and documentation is another heuristic with negative values as shown in Figure 10. The respondents state that is difficult to find help and documentation in Petcha.

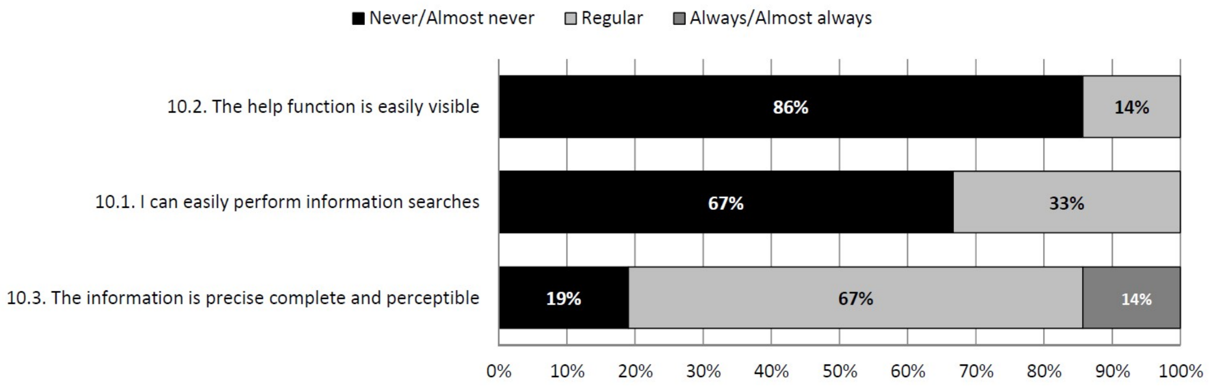

***Figure 10** – Results of each heuristic in the student profile.*

The respondents reveal that the error messages are sometimes unclear and inadequate in Petcha. The respondents also state that the documentation is scarce and is hard to find it.

The final classification of Petcha is shown in Figure 11.

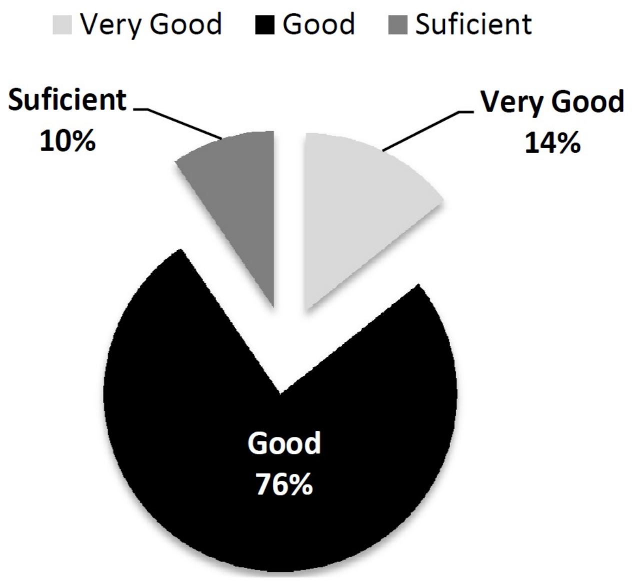

***Figure 11** – Classification of Pectha by students.*

One can conclude that the majority of students classified Petcha as a good tool according to the parameters evaluated.

Regarding the teacher profile, Figure 12 shows the results obtained.

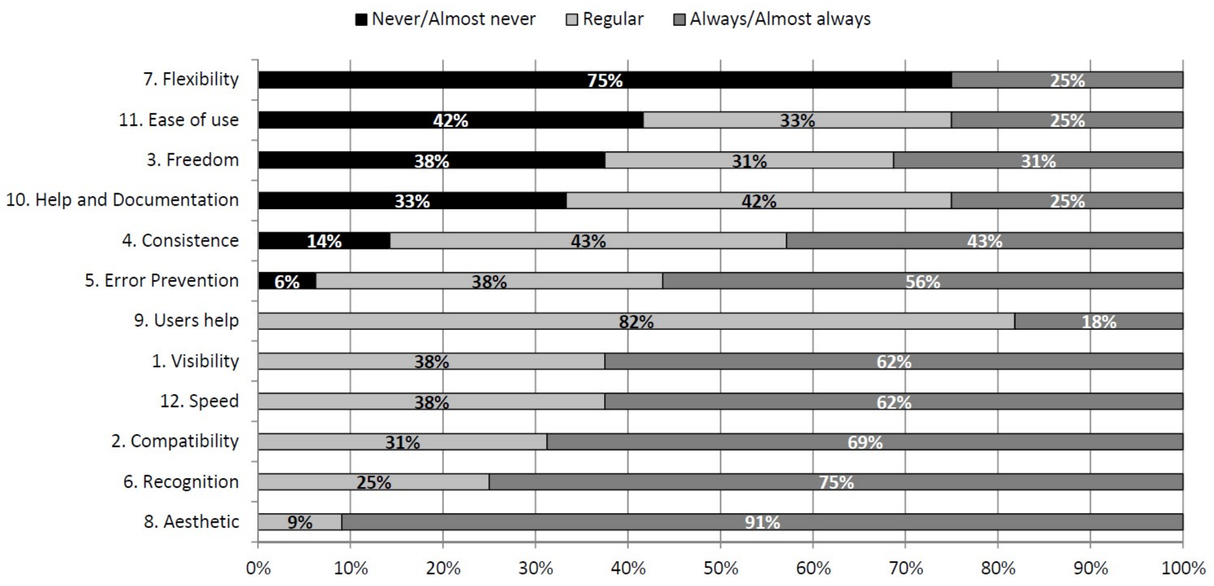

Figure 12 *– Results of each heuristic in the teacher profile.*

The results are similar to those of students. In one hand the aesthetic factor was the one with higher values of satisfaction. The respondents considered that the information contained on the screen is only what is needed and the system is aesthetically pleasing on the factors: color, brightness, etc. In the other hand the flexibility, freedom and documentation heuristics had some of the lowest values. However another heuristic had a low value: the Ease of Use. Figure 13 shows the results associated to this heuristic.

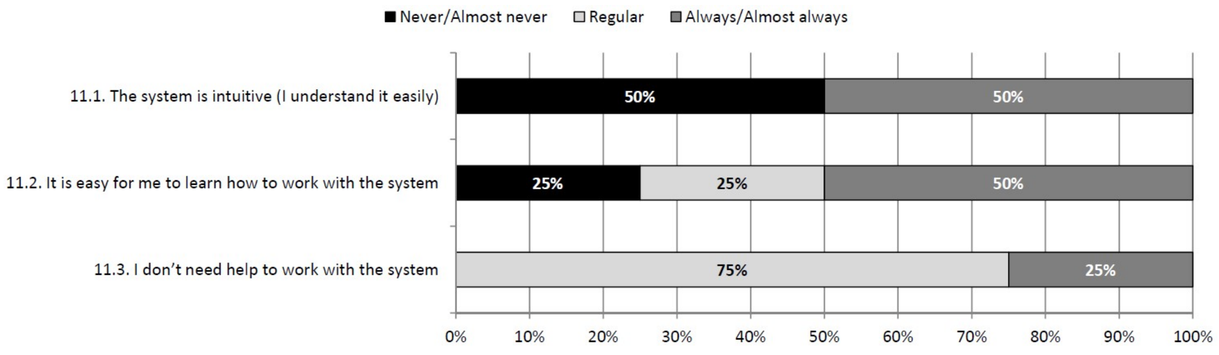

Figure 13 *– Results of the Ease of Use heuristic in the teacher profile.*

Reliability

The reliability of a system is the ability of a system or component to perform its required functions under stated conditions for a specified period of time.

In the final survey a reliability section was added in order to check on what tasks students and teachers had the difficulties. Figure14 shows the results of the survey in this facet.

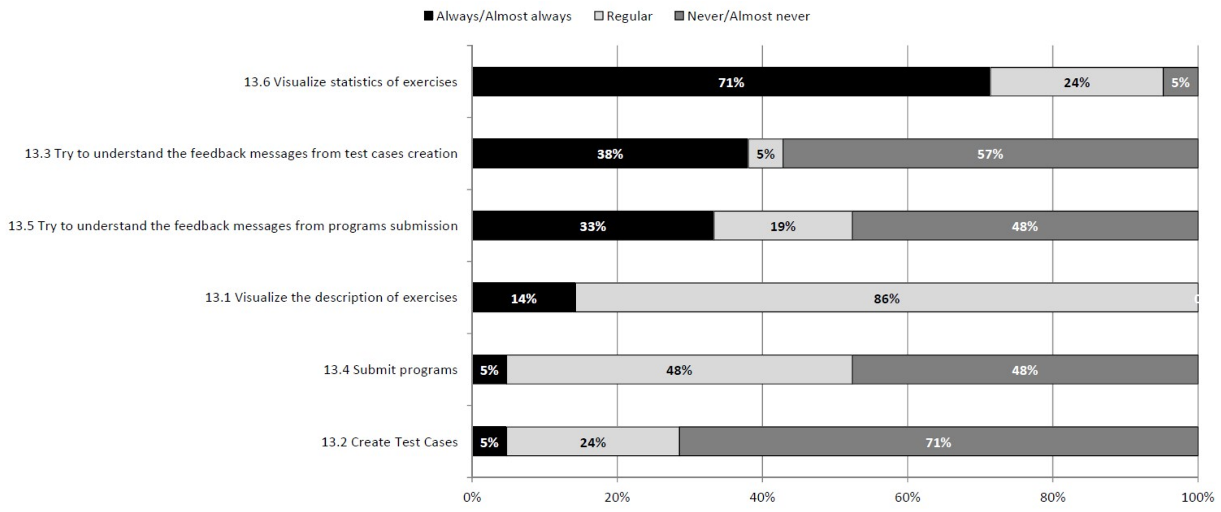

Figure 14 – *Reliability of Petcha.*

Students reported difficulties in understanding error messages related to programs submission. Some users have left comments and suggestions, and the following were the most common:

- the message wording should be more friendly;
- the feedback should be more complete;
- the help system is insufficient for novice users;
- the exercise statistics should be more complete (e.g. time to solve, students rankings).

The comments and suggestions of the teachers were the following:

- the creation of exercises is not intuitive;
- the evaluation based on test cases should be more flexible in the comparison between the outputs generated by the student solution with the accepted outputs;

\end{itemize}

Interoperability

In the design of the experiment a network was deployed based on the Ensemble instance previously. This network is composed by several systems and services that need to interoperate to achieve a common goal. In this subsection the interoperability of this network is validated. Table 1 summarizes the communication between pairs of systems. These results were gathered from 6 lab sessions. In each session a class of 21 students had 3 exercises to solve. Based on these figures were computed the following expected values: the expected number of accesses to the system that is given by multiplying the number of students with the number of assignments (21 * 6 = 126); and the expected number of accesses to the exercises by all the students that is given by multiplying the number of students with the number of assignments and with the number of exercises by assignment (21 * 6 * 3 = 378).

Observation Point	Expected	Real
# accesses to the system (LMS to TA)	126	135
# exercises requested to the repository (TA to LOR)	378	345
# exercises that students try to solve (TA to IDE)	378	342
# submissions (TA to AS)	378	819
# exercises requested to the repository (AS to LOR)	18	18
# exercises in which the students got feedback (AS to TA)	378	810

Table 1. Statistical data on interoperability of the network components

The first line of the table indicates that the system worked well since only nine extra sessions were used. These extra values were mainly due to the accidental closing of the application by students.

The number of exercises requested by the TA (Petcha) measures the number of times that students got an exercise statement from the repository. This action triggers an automatic request from Petcha to the repository. From the collected data it can be observed that not all exercises were actually read by all the students. There are two possible justifications: either the students did not have time to read all the available statements or some students may have given up after reading the exercise title. In any event students did not report any difficulty in accessing to problems

The third line of Table 1 is the number of exercises that students tried to solve. It is assumed that a student tried to solve an exercise when (s)he ran locally a set of tests to validate the code. The real number is less than the expected and less than the number of exercise statement accesses. Most likely some students read an exercise statement, but did not have time to code a solution, or run the tests, or simply given up on solving it.
The number of submissions is the number of requests for evaluation received by the AS. On average each student made approximately two submissions per exercise.

The number of exercises requests reflects the need of the AS to obtained the full LO from to repository given its reference. Since the AS has a cache mechanism the expected and real values are identical thus showing that the AS cache feature is working as expected and is accelerating the evaluation process.

The number of exercises in which the students got feedback should be similar to those of the number of submissions. Since they are almost identical (a difference of 9) one can conclude that the communication among the two systems (TA and AS) works well.

CONCLUSION

Learning complex skills is hard. Introductory programming courses are generally regarded as difficult and often have high failure and dropout rates (Ala-Mutka, 2005), (O'Kelly, 2006), (Robins, 2003). Many educators claim that "learning through practice" is by far the best way to learn computer programming and to engage novice students (Gross, 2005), (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, 2004).

This work proposes an e-learning framework - called Ensemble - that acts as a conceptual tool in the definition and deployment of e-leaning networks using complex evaluation. The framework relies on interoperability standards and specifications, thus several studies and surveys were conducted to select the most relevant for the framework. Based on this framework a network of systems and services was created and deployed for a specific domain - the computer programming domain. Content issues are tacked with a standard format to describe programming exercises as learning objects. Communication issues are addressed with the development or adaptation of systems and services for managing the life-cycle of exercises, namely their authoring, storage, conversion and assessment.

The framework instance was deployed for use in practical classes of undergraduate programming courses. The experience gained using Petcha (the pivot component) in this context and the experiments designed to assess the impact of this tool were also presented in this dissertation. These experiments showed an increase on exercises solving, attendance and grades when Petcha replaced a human Teaching Assistant (TA). However, these results show also that the automatic feedback provided by Petcha is less effective than that of a human TA. There is clearly room for improving automatic feedback in Petcha, although it can be argued that automated feedback is still a remedy for situations where a human TA is not available.

Contributions

The main contribution of this work is a conceptual model - the Ensemble framework - for the definition and deployment of e-learning networks using complex evaluation. The architectural model of this framework relies on central components (called axial systems) replicated for each teacher and student machines. One of these central components assumes a pivot role orchestrating all the communications within a single deployment of an Ensemble instance. Since it is distributed over each network user, this approach prevents any single-point-of-failure issues that might occur. This pivot component communicates locally with other axial systems and remotely with core and secondary services representing a distinctive feature regarding other e-learning frameworks.
Part of this main contribution is the specialization of the framework for a specific domain - the computer programming domain. This framework instance comprises several systems and services and their integration poses interoperability issues at two levels: content and communication. The content interoperability relies on the definition of an interoperability language for programming exercises called PExIL. The communication among systems and services was supported by the extension of existing specifications (e.g. IMS DRI, IMS LTI) and the creation of new ones (e.g. Evaluate service).

Other important contribution is the systematic study on the state of the art regarding e-learning systems and standards. This comprehensive study focuses on several surveys presented in the state of the art that were instrumental to chose the better e-learning systems and standards for the Ensemble framework. This contribution may prove helpful to other researchers studying the interoperability of e-learning systems.

The remaining contributions are related with the design and implementation of components that comprises the Ensemble network for the computer programming domain. These components are the crimsonHex repository (including a plug-in for accessing crimsonHex based repositories from Moodle), the Petcha teaching assistant and the BabeLO exercises converter. All these components are open source and can be downloaded from the following URL: http://ensemble.dcc.fc.up.pt.

Opportunities for future work

The motivation for this research was drawn from the computer programming domain. However, it was always kept in mind that the proposed concepts and tools could be used in other domains. The main opportunity for future research comes from extending this framework to other domains and requirements. Nevertheless, the evaluation of Ensemble and the validation of the thesis highlighted a number of issues that must be resolved in the computer programming instance of Ensemble.

Framework validation

The main challenge resulting from this research is to apply the framework to other domains. Although the research hypothesis were in general validated, the framework as such is not yet validated since it was only applied to a single domain. One interesting domain is serious games applied to management courses where students develop their skills using simulation. Business simulation games improve the strategic thinking and decision making skills of students in several areas (e.g. finances, logistics, and production). Through these simulations students compete among them as they would in a real world companies. A business simulation service fulfils a role similar to that of the assessment systems in programming exercises and it also requires a repository containing specialized LO describing simulations. Thus, this specific domain poses challenges not only in the development of the network TA, but also in the refinement of the framework specifications and services (e.g. repository, assessment system) to meet the new evaluation domains requirements.

Framework extension

The current version of the framework focuses mainly on exercise authoring, exchange and evaluation. Other kinds of data and services should be added to improve the practice-based environments supported by Ensemble, both in the computer programming domain and in new domains.

- Plagiarism checker - this component can be added to the framework to avoid plagiarism and ensure good scholarly practices. This tool is transversal to several areas and is therefore a good candidate to integrate the Ensemble framework;
- Sequencing component - 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 network pivot component will report the exercise assessment to this new tool that will use it to propose the appropriate content or exercise to the student;

- E-Portfolio - this is a special type of a repository where a collection of electronic evidence is assembled and managed by a user. They are distinct from LMSs since they are user-centric rather than course-centric. The integration of such tool in the Ensemble framework can be achieved at content or communication level through data (e.g. LeapA specification) or tools integration (e.g. IMS LTI specification);
- Standards and specifications - support for other LO package specifications (e.g. SCORM objects and for MathJax for displaying math expressions in the description of exercises.

Ensemble instance improvements

The computer programming instance of Ensemble is currently being used in the practical classes of undergraduate programming courses at ESEIG and will continue to be used in the next academic year. Several improvements are planned for immediate implementation based on the suggestions of teachers and students after the experiment. These improvements focus on Petcha - the visible system of the network, and include:

- User interface - make the GUI more intuitive and flexible;
- Evaluation reports - improve the visualization of the evaluation reports using new formats (e.g. PDF);
- Statistics - improve statistical data on student activity (e.g. time to solve, rankings);
- Help - extend the documentation to guide users;

In general, the improvements presented previously are minor issues that should be easily fixed for the next version of the Ensemble instance. There is also a collection of new features that would improve automatic assessment but that will require a major redesign of the AS.

- Feedback - improve the feedback mechanism based on, for instance, the use of static analysis over the students' code. Existing work in this area (Nielson, 1999) can be used to improve the feedback given to students after submission.
- New evaluation models] - a programming problem definition must have an unambiguous evaluation model. Typically a program from a student is assessed by the evaluator as a single program. Another approach is the student code be included within a set of programs from different learners for competitive evaluation. A third approach is where several programs from different learners are evaluated simultaneously interacting with a central component (an "oracle") also in a competitive fashion. For instance, in the Tic Tac Toe game the student's program plays the game against the oracle;
- Other types of languages] - the current evaluator can be configured for any programming language with a command line interface and processing standard input/output. There are computer languages that are not strictly programming but are regularly used in computer science courses such as query languages (e.g. SQL), modelling languages (e.g. UML) and user interfaces (e.g. HTML). In most cases these languages can be evaluated statically by comparing the submitted source code with the solution.

REFERENCES

Aguirre, S., Salvachua, J., Quemada, J., Fumero, A., & Tapiador, A. (2006, November). Joint degrees in e-learning systems: A web services approach. In Proceedings of the 2nd IEEE international conference on collaborative computing: Networking, applications and worksharing (collaboratecom 2006). Retrieved from http://jungla.dit.upm.es/ saguirre/publications/CollaborateCom20061:pdf

Ala-Mutka, K. (2005). A survey of automated assessment approaches for programming assignments. Journal of Computer Science Education, 15 (2), 83-102. (http://www.tandfonline.com/doi/pdf/10.1080/08993400500150747)

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)

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

Apostolopoulos, T. K., & Kefala, A. S. (2003). An e-learning service management architecture. In Icalt (p. 140-144).

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

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.

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)

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

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

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.
Digital Library Federation. (2007, September). Metadata encoding and transmission standard: Primer and reference manual. http://www.loc.gov/standards/mets/mets-schemadocs.html.

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

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

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., B   orstler, 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.

Gilbert, T. (2010). Leveraging sakai and ims lti to standardize integrations. In 10th sakai conference. 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., & 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. (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. (2010). From eLearning Systems to Specialised Services. Chapter of EduJudge project book called "A New Learning Paradigm: Competition Supported by Technology". Sello Editorial.

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

lok Lee, F., & Heyworth, R. (2000). Problem complexity: A measure of problem difficulty in algebra by using computer.

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). crimsonHex: a learning objects repository for programming exercises.. Softw., Pract. Exper., 43, 911-935.

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). BabeLO - An Extensible Converter of Programming Exercises Formats, IEEE Transactions on Learning Technologies, vol. 6, no. 1, pp. 38-45.

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.).

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

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

Rongas, T., Kaarna, A., & K alvi ainen, H. (2004). Classiffication 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

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., Miklos, 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)

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

Tremblay, G., Guérin, 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.

Vansteenkiste, M., & Deci, E. L. (2003). Competitively contingent rewards and intrinsic motivation: Can losers remain motivated? Motivation and Emotion, 27 , 273-299. Retrieved from http://dx.doi.org/10.1023/A:1026259005264 (10.1023/A:1026259005264)

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

E-Learning framework: A service-orientated approach to the development and integration of computer systems in the sphere of learning and education. Usually these types of frameworks come in two level: abstract and concrete.