Applying an LMS to Large Language Classes

Philip Grew*, Ivan Longhi^, Fiorella De Cindio*, and Laura Anna Ripamonti*
* Università degli Studi di Milano, Italy
^ Fondazione Rete Civica di Milano, Italy

Abstract:
Having applied an open-source LMS to about two thousand on-screen English exams for required courses that are part of the Università di Milano's computer-science short-degree program, the authors report on some of the interesting effects of using such a system. Large class size led to the adoption of an LMS designed for distance learning as a platform for automated test marking. A greater sense of belonging among students has resulted. Upon completion of a test, students immediately see where they stand vis-à-vis the group at large, and students who have not attended class can be brought into the system/community.

Keywords: *learning management system, LMS, on-screen testing, online teaching*.

Corresponding author: *Philip Grew; Dipartimento di Informatica e Comunicazione, Università degli Studi di Milano; Via Comelico 39/41; 20135 Milano, Italy; phone +39-02-502-16-320, fax +39-02-700-443-105, email grew@dico.unimi.it*

1. Introduction

English as a foreign language is a required course for some 800 first-year college students in three three-year degree programs in computer science at the Università degli Studi di Milano, Italy. Informational documents, syllabi, reading assignments, and the slides used in the classroom during these courses have been online for some time. In addition, forums had already been used both for general online discussion outside of class and for students' written projects. However, because class size remains unwieldy, even when students are divided into as many as six sections, a more efficient means of automating student participation, distributing course materials, and, especially, organizing exams was sought.

A solution was found thanks to research ties between the university's Dipartimento di Informatica e Comunicazione and the Fondazione Rete Civica di Milano, a foundation that runs a huge civic network in Milan (RCM) where an open-source learning management system (LMS) is under development as part of a project to support small and medium enterprises in their efforts to adopt emerging technologies (see [1], [2] and [3] for discussion). Thus, the English-language program was able to arrange to be among the early trial groups to use this LMS. The cornerstone of this non-profit project, based on the premise that learning takes place as a function of community (see [4] and [2]), involves distance learning on an LMS designed for the purpose and known as "JLI!" – an acronym of "Just Learn It!". In the case of the English-language courses, JLI! is being used primarily (but not exclusively, *vide infra*) as a support framework for bricks-and-mortar teaching, where it has been applied to approximately 2000 on-screen English exams.

Before this system was adopted, these tests were given on paper, either using Chemware QuizIt to scramble and print the questions and then analyze the scanned-in answer sheets, or, more commonly, simply hand-corrected. In the case of manually corrected tests, multiple versions nearly always had to be employed because the size of classrooms limits the number of seats that can be left empty. The tests being given are mostly at lower intermediate level and the material relates to standard English-as-a-foreign-language textbooks or, in some cases, English texts specific to a computer-science curriculum. While paper tests managed with QuizIt were necessarily multiple-choice only, the typical paper test used in these English classes has questions of many kinds, including fill-in-the-blanks, open answer, and so forth. The resources available to the English faculty members thus used to force a choice between tests that required a great deal of time and effort to design and correct or automated testing that severely limited our freedom of test design. The proximity of both computer labs and computer research made turning to colleagues in the department a logical step.

2. Development of the LMS known as "JLI!"

At the same time, the RCM Foundation was deciding to develop an LMS in house after a variety of software solutions had been considered and found wanting for various reasons. In particular, one of RCM's requirements was the integration of several community tools that were considered indispensable to the small- and medium-enterprise project. The JLI! e-learning platform was thus developed starting from the Adept (http://adept.sourceforge.net) open-source project, despite the fact that the latest available version was something less than a beta release.

Designed to be an upgradeable open platform (see [5] for fuller discussion) that easily integrates with other applications or services, JLI! is platform-independent and runs in an Apache/PHP environment, using the MySQL DBMS. The architecture of the Adept prototype was

completely redesigned so as to allow integration with other, external, platforms. For example, the discussion forums feature in JLI! simply implements a log-on interface with an external community application. The database structure was also modified in order to assure easy and flexible integrability through an LDAP directory, for example. JLI! runs handily on even a modest Unix Web server with up to at least about 100 students logged in at a given time (with the practical limit more likely to be determined by the size of the audio and video files and available bandwidth rather than server resources).

3. Design features of JLI!

Three characteristics of JLI! that made it well suited to the testing task that the English instructors envisioned were its dual view, integrability, and simplicity. By dual view, we mean the two ways in which a typical interactive learning object is displayed, with certain fields, i.e. question content, editable in the teacher's view but read-only in the student's view.

Integrability was important in the case of our on-screen English tests because it eased the transition from paper testing by allowing reuse of existing unique user accounts from the class forum based on an OpenText FirstClass server that most students already had, thus speeding the test-enrollment process.

While it is fair to expect a certain degree of computer-literacy among teachers now days, simplicity of use was an important consideration in that only a browser is required to access JLI! and no special programming skills are demanded of learning-object creators. The formating for quiz modules follows the style sheets applied to a given installation of JLI! A few special needs, such as having the student's name appear in large type at the top of the screen during student view, were thus easily applied ad hoc by the JLI! development team.

JLI! thus provided the English teachers with a WYSIWYG interface to create HTML contents that are immediately accessible to specified students who logged onto the systems using a Web browser. The platform allows teachers to upload and/or link any other resource they deem useful, including audio or video files, word-processor documents, on-screen presentations, image files, etc. It provides tools for the guided creation of FAQ and glossary sections, as well as a rich-text window in which even an English-teacher can quickly create an attractive homepage for each course she or he prepares on this system. When a student logs on, he or she first sees a list of "courses" to which access has been assigned. Each of these courses then opens with its particular homepage. Also unique to each course is a simple specification sheet that the teacher fills out in plain text and can be called up by the student in a read-only, popup window. Each English test set up on JLI! was included in one "course" in JLI! terms.

The basic building block of a JLI! course is the module. A given course may contain any number of modules and these may easily be copied from one course to another. Within a given course, the teacher has the option of leaving the student the freedom to view or complete each module in any order she or he chooses or forcing the student to proceed through the modules in the specified order. A module may contain any number of items of the type enabled by that module, i.e., in the case of an interactive module, any number of questions.

There are two categories of modules in JLI! for a current total of ten types of module that a teacher can choose to create when putting together a JLI! course. Three of these are broadcast modules, meaning that they consist simply of materials to be viewed (or heard) by the student. These are referred to as "theoretical" or "concept" modules in JLI! terms and essentially provide a way of creating reusable learning objects or managing existing didactic content. One type of theoretical module allows the teacher to create a page in rich text or HTML, with or without images inserted, using the included rich-text editor if so desired. Such a page then becomes a resource available on the system to other courses that teacher manages.

The second type of theoretical module is based on the idea of using multiple files, perhaps materials that the teacher has saved from previous classes. A presentation in Microsoft PowerPoint might thus be paired with an explanation written in word-processor format. Files can be of any type for which there are browser plugins and their content is loaded inside the JLI! Web interface. The third and final type of theoretical module allows for the display within the LMS window of a linked Web page or pages, using LDAP authentication if so desired (i.e. in the case of protected content).

The category of interactive modules comprises seven types of modules that are completed by the student taking the course (or, in our case, taking the test). Three of these are designed to produce results that are analyzed manually by the teacher: the open-answer, uploaded-file, and Likert-quiz modules. In an open-answer module, for example, a student might be asked to describe an attached photograph and given an electronic blank sheet on which to do it. Upon completion, when the student hands in the module by clicking on the "done" button, the module becomes read-only for the student and appears as awaiting correction in the teacher's view of the course (with possible email notification). When the teacher has viewed the module, she or he gives it a grade expressed as a percentage and, once these results too have been saved, the grade and any comments become visible to the student. The grade also appears in the result-summary windows (where each student sees his or her results for each course and the related class averages, on the student side, and where the teacher has a variety of views, including one with scores for all students on all modules, on the teacher/administrator side of the system).

4. Applying JLI! to the English tests

It should be stressed that this sort of module, as we have used it, i.e. with the students doing their typing while physically present in a computer lab and under the watchful eye of the teachers, actually adds nothing new to the testing process as compared to assigning a paper-based essay question. In intellectual and theoretical terms it should be identical. However, certain differences in the intangibles of testing were noted in psychological terms. Our subjective impression is that sitting at a keyboard tends to make students write more formally, despite the element of play that comes with the use of a computer. The effect of an automatic timing mechanism forces respect for the clock and prevents anyone from asking for "just a minute" more on a given module.

The greatest difference in the case of open questions between hand-corrected paper tests and manually graded on-screen tests has proven to be in the elimination of handwriting. Student handwriting is of varying quality, the cursive used by students of one culture differs from that used by the four teachers involved (who are all native speakers of English), and, most important, on a language test, a single letter can often make the difference between a right or wrong answer. The precision of the keyboard is well-suited to solving these problems.

The four other types of interactive JLI! modules are designed for immediate-response evaluation. One of them, the hidden-answer module, allows for the advance preparation of model answers to open or essay questions. This is best suited to the actual application of JLI! to its core purpose as a distance-learning platform for delivering educational content. Though it has not been used extensively in the language-testing project under discussion here, several other teaching projects are underway or planned that will test JLI! for language courses, as well as other subjects. The course content so far conveyed on this platform as part of these particular English courses has not been designed for distance learning or remote self-access. It is worth noting, however, that class attendance is not mandatory at Italian universities and that many students actually do not attend, often because they also have full- or part-time jobs. As a result, we have been able to ascertain that our online content is indeed popular as distance-learning material, as well. Once an examination "paper" has been retired, we make it available, upon request, to students who cannot attend class. We then often "see" these students virtually on the list of those currently logged on to the system.

The first-year, computer-science, English courses have relied most heavily on the three JLI! module types that provide for machine grading of students' test results. These three types of module are multiple-choice, true-or-false, and cloze (fill-in-the-blanks). Because a JLI! module may contain an unlimited number of questions, it would theoretically be possible to write an entire examination as a single module, if only multiple-choice questions were used. Because incorporated learning objects are linked to the individual questions rather than to the module as a whole, separate listening-comprehension passages can easily apply to different questions. In practice we have tended to break down the tests into sections even when several test sections were technically similar in that they relied on multiple choice. One reason for doing this is that it enables results to be analyzed and weighted according to the various skills being tested. Thus we are able to have separate grades for each student in general grammar, vocabulary, verb tenses, and so forth. A further reason is that a conceptual division makes the test interface more intuitively usable. This is especially so because, for examination purposes, we have always availed ourselves of the feature that allows the order of the questions on a given module to be randomized and the order of the distracters for each question to be jumbled. This is done to hinder efforts to cheat.

Early versions of JLI! showed that the multiple-choice module type could easily apply to true-false test sections as well, given that JLI! places no system limit on the number of distracters a multiple-choice question may have (although the current version employs a user-side interface designed for no more than ten). However, because one of the main advantages of true-false questions is the number of them that a student may be expected to digest in a given amount of time, it was deemed advisable to optimize the management of these questions by designing a specific module for them. This makes it possible to have any number of short reading passages directly in the test window, each followed by several statements to be marked true or false.

All the teachers involved in the project have made extensive use of cloze modules. Unlike some other quiz software, JLI! is not case sensitive and can accept more than one possible correct answer for each blank, which allows for creativity and leeway in test design. As with open-answer questions, the fact that student replies are typewritten overcomes one of the major drawbacks of this type of test question when used on paper-based exams. On a JLI! cloze-module question, when more than one correct answer has been programmed into the test by the teacher, the correction provided by the system to students who fail to answer a question correctly is the first of the correct answers programmed for that particular question.

All the machine-graded JLI! modules allow the teacher to choose whether, as the student completes and hands in the module, the corrections are shown on screen. In the case of on-screen testing, this option is naturally turned off while the examination is in progress. After the test is over and results have been checked for anomalies, turning on the feature means that students can see their results simply by logging in to the Web site.

5. How teaching and learning have been affected

Experience with cloze questions on paper-based exams had taught us that some acceptable alternative answer that had not been foreseen by the teachers writing the test often arises when hundreds of students take the test. As a result, one function built into JLI! was re-correction. This means that the teacher can, after the fact, add to the list of possible correct answers and recalculate results for the entire class. This function proved so popular that it has since been extended to the other machine-graded modules, as well. This essential change in the correction process, as compared with paper-based testing, has greatly increased teacher confidence in new questions, thus allowing for greater variety in the specific points raised on the exam.

Because the LMS itself was also under development and subject to change, many of the suggestions for added features or improvements that teachers using the platform came up with were immediately implemented. A sort of virtuous cycle was thus created in that the automatic re-correction feature led to greater use of experimental test questions, while increased readiness to try out new materials in turn led to the incorporation into JLI! of statistical-analysis tools that facilitate the review of student responses to cloze questions. Once teachers are used to having such options available, it becomes attractive to plan for dynamic correction of fill-in-the-blanks questions. Students using the platform for a second or third time become aware of this and have an added incentive to look into how a marginally correct answer might be deemed acceptable.

By encouraging students to go over their machine-graded results and write forum messages requesting consideration of any alternative answers they felt might be acceptable, we have achieved return rates of up to 100%, i.e. all the students at some exams have visited the site and looked up not only their overall results but the corrected modules as well. Because JLI! logs such activity, we can be certain that effort spent correcting tests and writing comments is not wasted. When we used paper tests, it was not unusual for the comparable return rate to be in single digits. On learning that they had failed an exam, few students would make the effort to show up during their professor's office hours so as to look over their mistakes.

From a teaching standpoint, the most remarkable effect of introducing on-screen testing, confirming Preece's observation of the social effects of platform usability [6], has been to foster community and participation. In part this is due to the more dynamic nature of the information, which has helped capture student attention, something teachers may be forgiven for considering an increasingly scarce resource [7]. Any tool that increases student interest is a welcome aid to teaching.

Surprisingly, attempts by students to pass answers to classmates have decreased rather than increased, compared to earlier paper-based testing. One incentive for such cheating among Italian students who do not previously know one another is clearly the sense of cooperating to oppose a common adversary, the instructor (or, at least, the institutional situation she or he embodies). This idea may be less felt during on-screen testing. In addition, the ability to see how one's score compares to a group average, which is a feature of JLI!, may alter the traditional climate of student solidarity. Finally, there is reason to believe that the shared online experience may at times give rise to an assumed code of behavior which might not discourage offering help to a friend but would make asking a stranger for it inappropriate [8].

While initial experimentation with English tests on the JLI! system involved midterm and final exams, our most recent efforts have extended the use of the system to start-of-term placement tests. It is interesting to note that even in this case, when students knew they would be informed of their test results in class a couple of days later, a high percentage returned to the online test to view their own "test papers" after testing had been completed. For example, this was done by 242 out of 400 students on the first round of entrance testing during the 2003-04 academic year. As word of this feature spread among the new students, the percentages returning to review their work rose to nearly 75%.

One result was that some new students showed up at the first day of class in the English section to which they had been assigned (on the basis of the test) with specific questions as to why a certain answer had been wrong on the placement test. The effect was to transfer online interactivity into the bricks-and-mortar classroom.

An additional aspect of the on-screen learning process when applied to entry testing can be seen in its effect in forging community. In part this results from the mere sense of shared experience that participating in a test day involving hundreds of new students inevitably spawns. And the novelty of on-screen testing probably contributes. But because entry-level computer-science students arrive with a vast range of computer skills, not to mention widely differing English-language abilities, we were able to watch newly acquainted classmates assist each other on their first day in the computer lab both in logging on to their individual completed tests and in understanding how a right answer distinguished itself from the other distracters, for example.

In the case of placement testing, the LMS medium provided a context for a group to form, as individuals, in Wenger's phrase, "engage in a process of collective learning that creates bonds between them" [9]. Our hope is that future efforts in Milan can be directed at extending this sense of community to students who do not actually attend class by providing them with a means to access and discuss learning objects designed for those who do.

Acknowledgments:
This work has been partially supported by the Italian Ministry of Education, Universities, and Research in the framework of the FIRB "Web-Minds" project.
The authors would also like to thank Prof. Patricia Cicogna for assistance in preparing tests administered on JLI! and Prof. Gian Paolo Rossi for supporting the experimentation.

References:

[1] M. Benassi, F. De Cindio, and L.A. Ripamonti, Network of Small Firms and the Web: Looking for a Trustworthy Architect, *Proc. of "IBMA 2003"*, Cairo, Egypt, 2003.

[2] F. De Cindio, I. Longhi, and L.A. Ripamonti, The First Need is a Learning Community: Just Do It!, *Proc. of IADIS International Conference e-Society 2003*, Lisbon, Portugal, 2003.

[3] L.A. Ripamonti, *Online Communities for Knowledge Sharing in SMEs*, PhD thesis, University of Milan, Italy, 2003.

[4] G. Casapulla, F. De Cindio, and L.A. Ripamonti, Community Networks and Access for All in the Era of the free Internet: "Discover the Treasure" of the Community, in L. Keeble and B.D. Loader (eds.), *Community Informatics: Shaping Computer-Mediated Social Relations* (London: Routledge, 2001).

[5] F. De Cindio, I. Longhi, and L.A. Ripamonti, Design Issues in Developing a Learning Environment For Small-sized Enterprise Communities, *Proc. of TEL03: Technology-enhanced Learning 2003*, Milan, Italy, 2003, 133-140.

[6] J. Preece, *Online Communities: Designing Usability, Supporting Sociability* (New York: John Wiley & Sons, 2000).

[7] Davenport, T. H., eLearning and Attention Economy: Here, There and Everywhere? *LINE Zine*, http://www.linezine.com/7.2/articles/tdeatae.htm, 2002.

[8] F. De Cindio, O. Gentile, P. Grew, and D. Redolfi, Community Networks: Rules of Behavior and Social Structure, *The Information Society*, 19(5), 2003, 395-406.

[9] E. Wenger, *Supporting Communities of Practice: a Survey of Community-oriented Technologies*, Draft Version 1.3, www.ewenger.com/ewbooks.html, 2001.