eWorkbook: an On-Line Testing System with Test Visualization Functionalities

Gennaro Costagliola, Vittorio Fuccella
Dipartimento di Matematica e Informatica, Università di Salerno
Via Ponte Don Melillo, I-84084 Fisciano (SA)
{gcostagliola, vfuccella}@unisa.it

ABSTRACT

On-Line Testing is that sector of e-learning aimed at assessing learner's knowledge through e-learning means. In on-line testing, due to the necessity of evaluating a big mass of learners in strict times, the means for knowledge evaluation had to evolve to satisfy the new necessities: *objective tests*, more rapidly assessable, started gaining more credence in the determination of learners' results. In this paper, we present an On-Line Testing system, named *eWorkbook*, which can be used for evaluating learner's knowledge by creating (the tutor) and taking (the learner) on-line tests based on *multiple choice* question type. Its use is suitable within the academic environment in a blended learning approach, by providing tutors with an additional assessment tool, and learners with a distance self-assessment means. Among other features, *eWorkbook* can record and visualize, in a suitable graphical format, learner's interactions with the system interface during the test session. This is valuable information for understanding the learner's behaviour when taking a test. In particular, the graphical analysis of the test outcomes has helped us in the discovery of several strategies employed by the learners to perform the test. In the paper, the main characteristics of the system are presented together with a rationale behind them and an outline of the architectural design of the system.

Keywords: Computer-Based Training, Distance Learning, Online Test, Web-Based Applications, Computer-Aided Software, Educational Technology.

INTRODUCTION

In blended learning the electronic means are mixed with the traditional didactics, in order to train and to assess the learners. *Learning Management Systems* (LMS), enhanced with collaborative environment support, and *On-Line Testing* tools are more and more widely adopted in the academy. At the University of Salerno some systems and platforms have been tested to support blended learning. Even if some good existing LMS with *On-Line Testing* capabilities, such as *Moodle* (Moodle, 2005) and *Sakai* (Sakai, 2005) have been used, none of them satisfied us at all: we needed an advanced assessment tool which could have helped the lecturers to speed up the onerous task of assessing a huge mass of learners and should have provided the tutor with valuable information for evaluating the whole assessment process.

A state of the art analysis undertaken at our department, which involved several lecturers and students, allowed us to identify the following important requirements for an effective environment for developing and using assessment tests:

- Item sharing features;
- Didactics organized in courses and classes;
- Possibility of administering both self-assessment tests and proctored laboratory exams;
- Availability of statistics on tests and questions;
- Availability of a rich reporting section on test outcomes.

A project for a comprehensive Web-based assessment system, named *eWorkbook*, was then started. The system can be used for evaluating a learner’s knowledge by creating (the tutor) and taking (the learner) on-line tests based on multiple choice question types. Even though *eWorkbook* allows the creation of on-line tests for both assessment and self-assessment, it was planned above all for *summative* (evaluation) purposes. The questions are kept in a hierarchical repository, that is, it is tree-structured, in the same way as the file system of an operating system. In such a structure, the files can be thought of as questions, whether the directories can be thought of as *macroareas*, which are containers of questions usually belonging to the same subject. Every item (a macroarea or a question) has an owner, which is the tutor who authored it. The tutors can choose whether to share their questions or not, assigning a value to the permissions associated to each item. Permissions are for reading, writing and using the items.

The tests are composed of one or more sections. This structure facilitates the selection of the questions from the repository, but it is still useful for the assessment, where it can be important to establish if one section is more important then another to determine the final grade for the test. The selection of the questions can occur both statically, by directly choosing the questions from the repository, and dynamically, leaving the system to choose the questions randomly.

Didactics are organized into courses and classes: the tutors responsible for a course manage its class and choose the tests that must be taken by the learners of that class. With such an organization, the system can be used by a large set of users, such as the learners and the tutors of an entire faculty. Within a course interface, the learner can easily access the self-assessment tests. Restrictions on the access rules can be defined for proctored laboratory tests.

Different assessment strategies can be bound to a test, before it is published in a course. The assessment strategy affects the way in which some parameters concur to determine the grade of the test. Some strategies are preloaded in the system and are referred to as *predefined assessment strategies*. Others can be defined by the tutors and saved in his/her reserved area. We will refer to them as *customized assessment strategies*.

A complete history of learners' performance on tests of the valuable list is available to the tutor and to the learners themselves. Each record in the history contains the date and the time when the learner has joined a test, the amount of time needed to finish the test and some information about assessment (test score and state). The detail of the answers to each question can be seen as well and can be viewed in a printer-friendly format. The tutor has the possibility of analyzing the behaviour of each learner during a test, by simply inspecting a suitable chart, which graphically summarizes learner's interactions with the system interface during the test session. By analyzing the data visualization charts from several test outcomes, we have detected several previously unknown test strategies used by the learners.

The rest of the paper is organized as follows. In the section *"The Main Features of eWorkbook"* the main features of the systems are described in detail. The section *"eWorkbook Architecture"* is devoted to outlining the architecture of eWorkbook. An example of system use can be found in section *"AnExample: The English Knowledge Test"*. In section *"Related Work"*, a comparison is made with some interesting systems related to ours. Some final remarks and a description of future work conclude the paper.

THE MAIN FEATURES OF EWORKBOOK

In the following subsections we will outline the main characteristics of the eWorkbook system. It is worth noting that eWorkbook was intended to be used by a large number of users, so it has a typical LMS didactics organization, based on courses and classes. A course is a place in which the tutors can publish tests and the learners can take them. Learners can only view the tests published in the courses in which they are members. The tutor manages the class and can accept or deny learners' affiliation requests and expel a learner from the course.

Question Management

An important matter for On-Line Testing, and more generally for e-learning, in order to accelerate the teaching and the assessment processes, is the reusability of the authored content. The on-line material needs a huge initial effort to be created, while it can be easily modified and reused later on. Therefore it is very important that existing material can be easily found, modified and selected by a tutor who wants to use it for a lesson or a test. There are two main ways to boost the reuse of learning material:

1. Good organization of material kept in an e-learning platform or On-Line Testing system.
2. Interoperability among systems and platforms, to share and exchange material.

Our system was designed to have a well organized question repository to facilitate the tutor in the question management, share and reuse: the question repository of eWorkbook has a hierarchical structure, similar to the directory tree of an operating system. Each item in our repository is a disciplinary *macroarea* (internal node) or a *question* (leaf). The membership of a question to a given macroarea is determined by its subject: each macroarea is a container of questions that holds items dealing with a specific subject. It can be further split in other sub-macroareas, which hold questions belonging to a more specific matter. The question types allowed are multiple choice, multiple response and true/false. The tutor can choose if a question should be used for assessment only, for self-assessment only or for both of them.

An effort for the interoperability has been made supporting the *IMS Question & Test Interoperability* specification (IMS QTI, 2005): our system can import and export information regarding questions and tests through this widely known and adopted XML-based format.

Permissions

Author's right protection is an important matter too. An e-learning system should offer the tutor the choice to share his own material or not. In eWorkbook, the owner (the tutor who authored the question) and a permission set are associated to each item. The owner establishes the values for each field of the permission set. A permission is a Boolean value that indicates whether other users beyond the owner can perform the action associated to that permission.

For a macroarea, the value for the following permissions must be set:

- *ReadPermission*: the permission to read the property and the contents of this macroarea.
- *WritePermission*: the permission to overwrite the property and manage this macroarea (add a sub-item to it, delete it).
- *UsePermission*: the permission to select a question from this macroarea for a test.

For a question, the permissions are the following:

- *ReadPermission*: the permission to read the question.

- *WritePermission*: the permission to delete and overwrite the question.
- *UsePermission*: the permission to select this question for a test presentation. Its default value is the value of UsePermission of the macroarea which this question belongs to.

It's worth noting that permissions are a good way to protect author's right and to avoid that the material owned by a tutor is modified or used without his/her consensus. Other systems only give the possibility to share or not all of the tutor's questions. A permission based system gives more flexibility to the system, allowing different grades of item sharing.

Question Metadata

Each question in the repository has a metadata set associated to it. Some of the parameters are decided by the tutor when he/she instantiate the metadata and they can be updated later, others are inferred by the system during its use. Inferred metadata are updated whenever a learner submits a test. Metadata are used in question selection in a way that will be clear in the sequel. The following is a list of the metadata fields:

- *Language*: the human language in which the question is expressed.
- *Keywords*: a set of keywords that describe the content of the question.
- *Use*: the aims the question is for. It can be *self-assessment*, *valuable* or *both.*
- *TestOccurrence*: an inferred field that is increased by one whenever this question is scheduled for a test.
- *AverageAnswerTime*: an inferred field. It can be used on our system because it is able to track the time spent by the learner on each question.
- *Difficulty*: this field has both an inferred and a tutor chosen value. It's a value between 0 and 1 that expresses a measure of the difficulty of the question, intended as the proportion of learners who get the question correct. The tutor can guess this value at the question creation time and can update it during the question's lifecycle. The system calculates the inferred value with a simple formula.

- *Discrimination*: this field is an inferred one. Its value is a measure of how well this question discriminates between learners. A good question should give full mark to good learners and penalize bad ones. Starting from this information, a great deal of criteria can be adopted. A solution is proposed in (Lira et al., 1990): it identifies a good question as the one which the better 20% of learners answers well and the worse 20% of learners answers incorrectly. We adopted a common solution applied in *Item Analysis*, calculating discrimination as the Pearson correlation between the score achieved on the question and the total score achieved on the test in which the question was scheduled. Its value is given by the following formula:

$$r = \frac{\frac{\sum (x-\bar{x})(y-\bar{y})}{n-1}}{\sqrt{\frac{\sum (x-\bar{x})^2}{n-1}}\sqrt{\frac{\sum (y-\bar{y})^2}{n-1}}}$$

  where the following rules are valid:

  - $-1 \leq r \leq 1$,
  - x is the series of the results got on the question,
  - y is the series of results got on the whole test.

Question Quality Improvement Through Question Lifecycle

In On-Line Testing systems it is important that the quality of the questions is kept high, so that the tutor can assess learners properly, using unambiguous questions that really distinguish between good learners and bad ones. eWorkbook adopts the statistical indicators (Difficulty and Discrimination, seen in the previous chapter) from *Item Analysis* to get information about the effectiveness of the questions.

The improvement of the quality of the question requires the use of a process which allows the tutor to analyze the entire lifecycle of a question, including all its previous versions and the learners' answers to them. Our question repository has a *Version Control System* that allows tutors to change some data of the questions, e.g. text, distractors or metadata, still keeping the previous versions of the question: the upgrade of a question does not imply the erasing of the previous version. This could be an important feature for reasons bound to the history of learner's responses to the question too: the question could already have been used in some tests before the upgrade, and the system has to remember which version of the question the learner answered. However, the *Version Control System* is important for reasons related to the quality of the questions: thanks to the tracking of the question lifecycle the tutor has feedback on the variation of statistical indicators over time. In this way, the tutor can modulate the difficulty of the question and make sure that the changes he/she made to it (maybe eliminating misspellings and ambiguities), affected positively the quality of the question. Other information, useful to establish the effectiveness of a question, is available: the tutor can easily inspect how many times it was selected to be presented in a test, the number and the percentage of correct, incorrect and not answered responses and the average time needed to get the response.

In the light of the previous arguments, we can argue that the definition and the use of questions from the hierarchical repository for more than one session of tests, combined with the *Version Control System*, allows the tutors to have a wide choice of high quality questions to select for their online tests.

Test Management

A test is composed of *sections*. eWorkbook has two ways of selecting the questions to be presented in a test: through a static creation-time choice or a dynamic run-time one. In the first case, the tutor has to choose the questions directly during the creation of the test; in the latter case, she/he has only to specify some selection parameters, letting the system choose randomly the questions across the chosen macroareas whenever a learner takes a test.

Therefore, we have two kinds of sections: a *static* section is an explicit selection of the questions to present performed at test creation time, while a *dynamic* section is characterized by a set of rules that perform a selection on the entire repository. For a dynamic section, there are three kinds of selection rules:

1. Definition of the macroarea. This rule limits the selection only to the questions belonging to the specified macroarea. A flag can be set, that further selects also the questions belonging to the descendent macroareas.
2. Definition of keywords. This rule limits the selection only to the questions that match the input keywords. Some logical connectors, in a search engine style can be used. By default, the questions which contain even one of the input keywords are selected. No relevance rate is associated to the results.
3. Definition of some assertions on metadata fields. They are of the following form: *<metadata_field> <connector> <value>*. As an example, for a section, we can choose to use only those questions that have *difficulty > 0.5.*

The same three rules are also used to search the questions for a static section through a wizard in the Web-based interface. The tutors can choose to use just one of them to select the questions, or to combine them to narrow or to expand the search results. The tutors can also choose whether to use only their material or the one shared by the other tutors too.

These rules allowed us to overcome problems related to question selection: different tests for each learner can be generated still getting an objective assessment through the selection of ranges for the difficulty and the average answer time. The discrimination was decided not to be used for question selection assertions, in order to avoid the neglecting of low quality questions. Our policy was to encourage the tutor to review low quality questions, in order to correct their anomalies and increase their quality.

Test Presentation

Two different lists of tests are presented to the learner within the course interface: the *valuable* and the *self assessment* test lists. Each test in the former list is used to determine the learner's evaluation and is characterized by an access control specified by a prerequisite expression and a maximum number of attempts. The latter list is just a guide for the learner to self train and assess: each test in it has not got any access restriction and does not affect the learner evaluation. Each test presented in a course is bound to some test execution options. These options allow the tutor to customize the test with further information which could not be available or decided at the test creation time, so we choose not to hard-code them in the test. Test execution options include the following information:

- *IP Limitation*: an option through which the tutor can authorize or deny access to some clients, according to their IP. A selection of authorized IP lists must be chosen. This option can be particularly useful for official exams, whose tests are required to be taken only by the learners that physically present in a laboratory. An IP list can be defined and selected for all the PCs of that laboratory. Wildcards and IP ranges can help to define IP lists.
- *Assessment*: a list of options that specify the numeric scale for the mark, the threshold to pass the test and the marking strategy. Details about marking strategies can be found in section *"Assessment Strategies"*.
- *Shuffle*: this Boolean option can be checked if the tutor wants to randomize the sequence of the questions, to make it more difficult for the learners to cheat.
- *Access Control*: this section of options is valid only for valuable tests. The tutor can choose the maximum number of attempts allowed for the test and the prerequisites for accessing it. Prerequisites establish, through a simple even powerful expression, the learner's right to access the test. If not fulfilled, prerequisites can deny learner's access to the test. Prerequisites for a test are based on the learner results on the previous tests in the valuable test list. The language supported for the expression is *aicc_script*; a string expressed in such a language has a Boolean value and it is composed of the following elements:
    - *Identifiers*: nouns that univocally identify a test in the valuable list.
    - *Constants*: values that define the state of a test (passed, completed, browsed, failed, not attempted, and incomplete).
    - Logic, equality and inequality *operators*.

- A special syntax to define a set and to specify at least n elements from a set.

  As an example: the expression *test1 & 2*{test2, test3, test4}* is true if the state of *test1* is *passed* or *completed* and at least two among *test2*, *test3* e *test4* are *passed* or *completed*. A simple visual interface helps the tutor to define the prerequisites string without knowing *aicc_script* language. There is also an *aicc_script-to-natural* language translator to help the learner to better understand the prerequisites for a test. A better and more complete explanation of aicc_script can be found in (ADL, 2001).

An instance of test execution options is a configuration that can be saved with a name and recalled in a second time, whenever a new test must be added. The tutor can opportunely define different sets of execution options in order to choose the most suitable configuration for a given test type (self-assessment, proctored laboratory exam, etc.).

Assessment Strategies

eWorkbook provides a wide choice of predefined assessment strategies and the possibility to define a new customized assessment strategy. An assessment strategy is a set of choices of the values to give to some parameters taken into account during the test assessment process. The predefined strategies are preloaded in the system and cannot be changed. They are at the disposal of all of the tutors. The customized strategies can be defined by a tutor, and they remain visible only in his reserved area. All the strategies calculate the final mark on the test by summing the results achieved on the single questions. The maximum mark which can be obtained on a single question depends on the weight of the question. A weight is assigned by the tutor to each section of questions in a test and the weight of a question is easily calculated by dividing the weight of the section by the number of questions in it. The customizable parameters are the following:

- *Weighting*: this parameter, if set, enables the *weighted assessment* for a test, that is, the maximum mark got on the question depends on its weight. If a tutor wants a section to be more important than the others, he/she has to give a higher weight to it during the test authoring, and he/she has to choose an assessment strategy with the weighting parameter set. If this parameter is not set, all the questions equally contribute to get the mark on the whole test.

- *BonusOnCorrect*: this parameter, if set, allows the tutor to specify a positive real factor (bonus) by which the mark obtained on the correctly answered questions during the assessment process must be multiplied.
- *PenaltyOnIncorrect*: this parameter, if set, allows the tutor to specify a negative real factor (penalty) by which the weight of the incorrectly answered questions during the assessment process must be multiplied. If not set, the mark obtained on the questions answered incorrectly is zero. It is possible to choose a *fair* penalty, which gives to the questions answered incorrectly a mark of *–(1/NC-1)*, where *NC* is the number of choices for a question. The use of the fair penalty should set to zero the mean mark for a question guessed by a learner who does not know the right answer to it.
- *PenaltyOnNotAnswered*: this parameter, if set, allows the tutor to specify a negative real factor (penalty) by which the weight of the unanswered questions during the assessment process must be multiplied. If not set, the mark obtained on the unanswered questions is zero.

Table **1** summarizes the values given to the parameters above for each predefined strategy. The names of the strategies have been taken from (IMS ASI, 2004). As we can see, for each strategy, there is a weighted version. None of the predefined strategies adopts bonuses on correct or penalty on not answered questions. *NumberCorrect* is a ‘plain’ strategy: none of the parameters is set. Its name is due to the way in which it calculates the mark on the whole test: just summing the number of corrected answers (and scaling the result to 30 or 100). *GuessingPenalty* and its weighted version *WeightedGuessingPenalty* use 1 as factor for the PenaltyOnIncorrect parameter. This means that they subtract the entire weight of the incorrectly answered questions from the final mark on the test. *GuessingFairPenalty* and its weighted version *WeightedGuessingFairPenalty*, use the fair penalty, explained above.

Table 1. Parameters of each predefined strategy

Strategy Name	Weighted	BonusOn Correct	PenaltyOn Incorrect	PenaltyOn NotAnswered
NumberCorrect	NO	NO	NO	NO
WeightedNumberCorrect	YES	NO	NO	NO
GuessingPenalty	NO	NO	YES (1)	NO
WeightedGuessingPenalty	YES	NO	YES (1)	NO
GuessingFairPenalty	NO	NO	FAIR	NO
WeightedGuessingFairPenalty	YES	NO	FAIR	NO

History Tracking

A complete history of a learner's performances on valuable test list is available to the tutor and to the learner him/herself. The tutor can view the results achieved by all the learners in his/her classes, while the learner view is restricted only to his/her results. Each record in the history contains the date and the time when the learner joined a test, the amount of time needed to finish the test and information about assessment (test score and state).

To consult the history, a search engine style form must be filled. The fields of the form allow the seeker to select a course, a learner and a test whose instances must be shown. Further advanced parameters, which allow to narrow the research, are: the state (terminated, not terminated) and the result (passed, not passed) of the test, a date range during which the test was taken, and the number of results per page.

Each instance present in the result pages has a link to a pdf file that contains a printable version of the test with all the learner's answers. A unique pdf file for all the instances is available as well. In such a way, all the tests can be saved or printed in one operation.

Test Visualization

eWorkbook tracks learner's interactions during the execution of a test. This can be useful for understanding the strategy used by the learner to complete the test and for giving him/her advise on how to perform better in future tests. Several experiments have been performed to this aim (Bath, 1967; Johnston, 1977; McClain, 1983) in the past. Our approach is the following: learner's interactions during tests are logged and stored in XML files; then the information gathered is analyzed and visualized in a suitable chart.

By showing the salient points of a test execution, a chronological review of the test is available to the tutor. The chart shows, at any time, the item browsed by the learner, the mouse position (intended as the presence of the mouse pointer on the stem or on one of the options) and the presence of response type interactions, correct or incorrect. The chart is two-dimensional: the horizontal axis reports a continuous measure, the time, while the vertical axis displays categories, the progressive number of the item currently viewed by the learner. The test execution is represented through a broken line. The view of an item for a determined duration, is shown through a segment drawn from the point corresponding to the start time of the view to the one corresponding to its end. Consequently, the length of the segment is proportional to the duration of the visualization of the corresponding item. A vertical segment represents a browsing event. A segment oriented towards the bottom of the chart represents a backward event, that is, the learner has pressed the button to view the previous item. A segment oriented towards the top is a forward event.

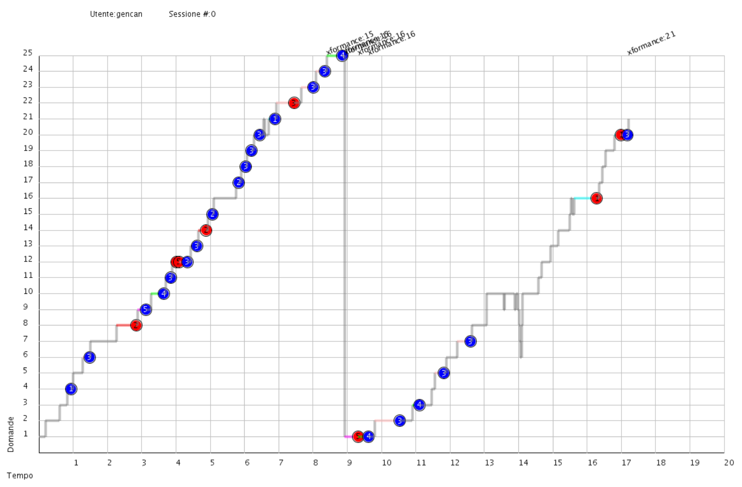


Figure 1. Graphical Chronological Review of a Sample Test.

The responses given by a learner on an item are represented through circles. The progressive number of the chosen option is printed inside the circle. The indication of correct/incorrect response is given by the filling color of the circle: a blue circle represents a correct response, while an incorrect response is represented through a red circle. The color is also used for representing the position of the mouse pointer during the item view. The presence of the mouse pointer in the *stem* area is represented through a black color for the line. As for the options areas, the red, yellow, green, blue and purple colors have been used, respectively, for 1 to 5 numbered options. More than 5 options are not supported at present. Lastly, grey is used to report the presence of the mouse pointer in a neutral zone. The graphical chronological review of a sample test is shown in figure 1. By analyzing the charts obtained with system use, we realized that learners often follow common strategies for completing on-line tests. In our experiments we have identified the following three mostly employed strategies:

- **Single Phase**. This strategy is composed of just one *phase* (a part of the test execution needed by the learner for sequentially browsing all or almost all of the questions in a test). The time available to complete the test is organized by the learner in order to browse all the questions just once. The learner tries to reason upon a question for an adequate time and then gives a response in almost all cases, since he/she knows that there will not be a revision for the questions. Eventual *phases* subsequent to the first one have a negligible duration and no responses.
- **Active Revising**. This strategy is composed of two or more *phases*. The learner intentionally browses all the questions in a shorter time than the time available, in order to leave some time for revising *phases*. The questions whose answer is uncertain are skipped and the response is left to subsequent *phases*. As a general rule, the first *phase* lasts a longer time and the subsequent *phases* have decreasing durations.
- **Passive Revising**. This strategy is composed of two or more *phases*. The learner browses and answers all the questions as fast as possible. The remaining time is used for one or more

revising *phases*. As a general rule, the first *phase* lasts a longer time and the subsequent *phases* have decreasing durations.

For both the definition of the strategies and the classification of test instances, the charts have been visually analyzed by a human operator. The above tasks are rather difficult to perform automatically, while a trained human operator can establish the strategy used by the learner from a visual inspection of the charts of the test instances and giving advice to the learners on how to perform better next time. Samples of the strategies are shown in figure 2.

Other uses of the above described method are the detection of correlation among questions and the detection of cheating during tests. The reader can refer to (Costagliola et al., 2007) for obtaining a more detailed description on the educational results of our experiments with the tool.

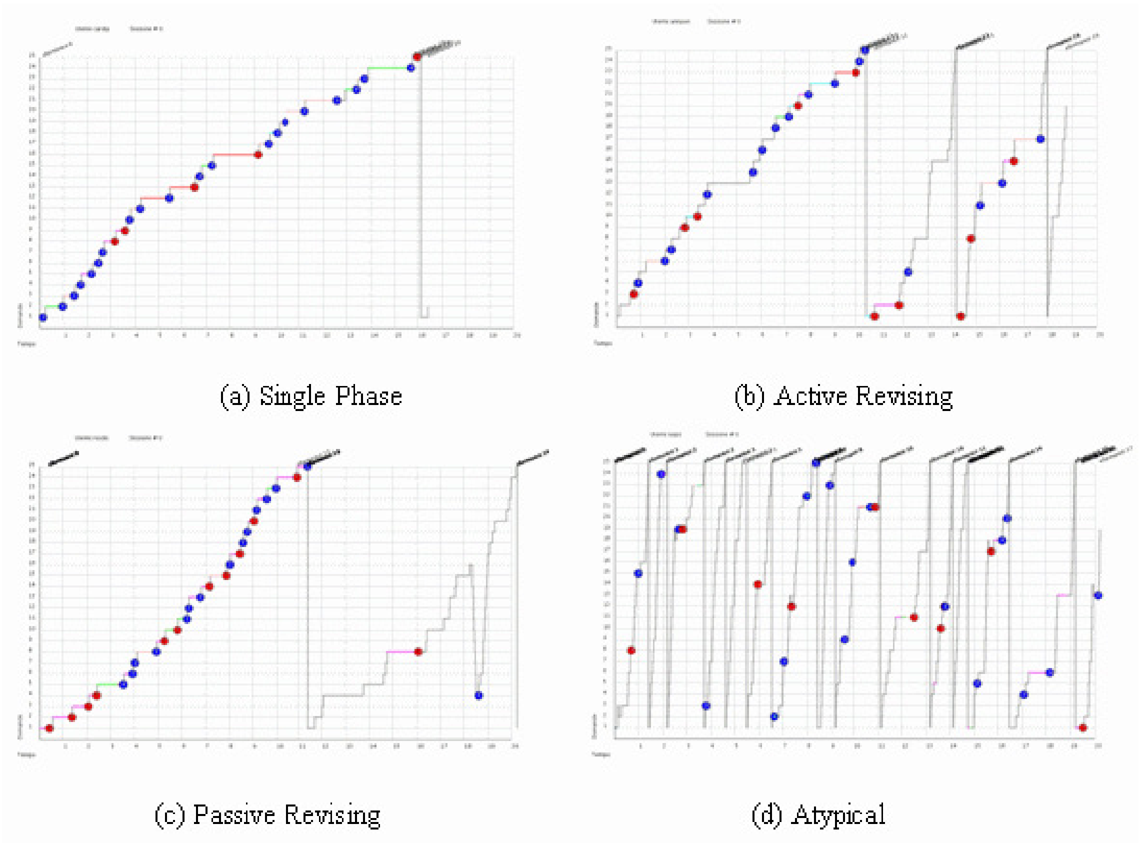


Figure 2. Sample of Test Execution Strategies.

EWORKBOOK ARCHITECTURE

As shown in Figure 3, eWorkbook has a layered architecture. The *Jakarta Struts Framework* (Struts, 2005) has been used to support the *Model 2* design paradigm, a variation of the classic *Model View Controller* (MVC) approach. Struts provides its own Controller component and integrates with other technologies to provide the Model and the View. In our design choice, Struts works with *Java Server Pages* (JSP, 2005), for the View, while it interacts with *Hibernate* (Hibernate, 2005), a powerful framework for object/relational persistence and query service for Java, for the Model.

The application is fully accessible with a Web Browser. Navigation is facilitated across the simple interfaces based on menus and navigation bars. User data inserting is done through HTML forms and some form data integrity checks are performed using Javascript code, to alleviate the server side processes. A big effort was made to limit the use of client-side scripts only to the standard *EcmaScript* language (ECMAScript, 2005). No browser plug-in installations are needed. It is worth noting that the system has been tested on recent versions of the most common browsers (i.e., *Internet Explorer*, *Netscape Navigator*, *Firefox* and *Opera*).

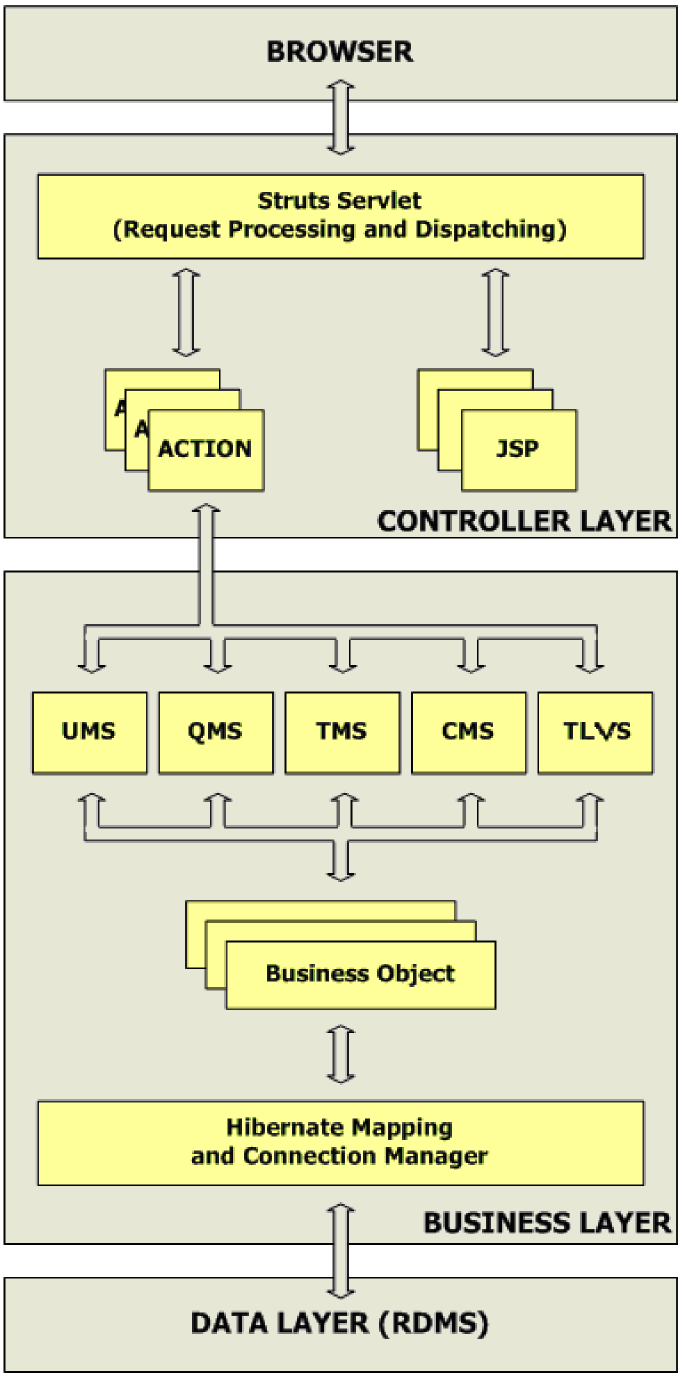


Figure 3 - Architecture of eWorkbook

The Web Browser interacts with the *Struts Servlet* that processes the request and dispatches it to the *Action Class*, responsible for serving it, according to the predefined configuration. It is worth noting that the Struts Servlet uses the JSP pages to implement the user interfaces. The Action Classes interact with the modules of the *Business Layer*, responsible for the logic of the application. The Business Layer accesses to the *Data Layer*, implemented through a Relational Data Base

Management System (RDBMS), to persist the data across the functionalities provided by Hibernate framework.

Controller Layer

This layer has many duties, among which are: getting client inputs, dispatching the request to the appropriate component and managing the view to return as a response to the client. Obviously, the Controller layer can have many other duties, but those mentioned above are the main ones.

In our application, following the Struts architecture, the main component of the Controller layer is the Struts Servlet, which represents the centralized control point of the Web application. In particular, the Struts Servlet processes each client request and delegates the management of the request to a helper class that is able to execute the operation related to the required action. In Struts, the helper class is implemented by an Action Class that can be considered as a bridge between a client-side action and an operation of the business logic of the application. When the Action Class terminates its task, it returns the control to the Struts Servlet that performs a forward action to the appropriate JSP page, according to the predefined configuration.

To reduce the effort to maintain and customize the application, we chose to limit the use of the JAVA code in the JSP pages, using as an alternative the Struts taglibs. In this way the Web designers are able to work on the page layouts without shouldering the programming aspects.

Finally, thanks to the use of the Struts framework, eWorkbook has the complete support for the internationalization of the Web-based interface. Even if, in its earlier releases, it only came with the English and Italian versions, the translation is quite an easy duty: to add a new language version all that our system needs is the translation of some phrases in a .properties (plain text) file. The Web pages are returned to Web browsers in the language specified in the header of the request.

Business Layer

This layer contains the business logic of the application. In any medium-sized or big-sized Web application, it is very important to separate the presentation from the business logic, so that the application is not closely bound to a specific type of presentation. Adopting this trick, the effort to change the look & feel of eWorkbook is limited to the development of a new user interface (JSP pages), without affecting the implementation of the other components of the architecture.

As mentioned before, every Action Class of the Controller Layer is able to execute an operation of the business logic of the application. To this aim, the Action Classes interact with four different subsystem of the Business Layer (see Figure 3). These subsystems are:

1. *User Management Subsystem* (UMS): this subsystem is responsible for user management. In particular, it provides insert, update and delete facilities.
2. *Question Management Subsystem* (QMS): this subsystem manages the question repository of eWorkbook and controls access to it. It is composed of two modules:
   a. *Question Repository Manager*: this module allows the management of the hierarchical structure of the question repository. Each internal node in it is a disciplinary macroarea, while each leaf is a question. This module allows the insertion, update and deletion of a macroarea and/or a question from the repository.
   b. *Access Permission Manager*: this module controls access to the question repository. For each node of the question tree it is necessary to specify the owner (i.e., the tutor who authored the macroarea or the question) and a permission set. The owner establishes the value for each field of the permission set.
3. *Test Management Subsystem* (TMS): this subsystem manages the test repository of eWorkbook. To achieve this, we have divided this subsystem into four modules:
   a. *Authoring Manager*: this module permits to create a new test, defining the questions that compose the test and the test execution options. The Authoring Manager also allows the publishing of an existing test in one or more courses;
   b. *Assessment Manager*: this module performs the test evaluation and manages the assessment strategies;

    c. *Execution Manager*: this module manages the test execution. To aim this, the Execution Manager gets a test instance from the Authoring Manager and performs the necessary operation to present it to the user. At the end of the test execution this module passes the control to the Assessment Manager to valuate the test;
    d. *History Manager*: this module manages the history of a learner's performance and a test's execution.
4. *Course Management Subsystem* (*CMS*): this subsystem manages the courses. In particular, it allows the insertion, update and deletion of a course.
5. *Test Logging & Visualization Subsystem (TLVS)*: this subsystem is composed of a *Logging Framework* and a *Log Analyzer*. The former is an already existing OO Java Framework which captures and logs all of the learners interactions with the on-line testing system interface and can be instantiated in any on-line testing system. It is further composed of a client-side, based on AJAX technologies, and a server-side module. The latter is a module that analyzes the logs in order to extract information from them and to graphically represent it.

It is worth noting that all the subsystems described above access to one or more business objects to manipulate information that is stored in the database. The Hibernate framework is used to manage those business objects that accede to the data layer across an appropriate mapping. The target of this mapping is to transform a relational database (stored in the data layer) in a light OO database; in this way it is possible to manage the data exploiting the advantages provided by the OO paradigm.

Data Layer

This layer contains the information stored in a RDBMS. It is worth noting that eWorkbook is not closely bound to a specific RDBMS, but supports much of the most popular RDBMS (i.e., *MySQL* (MySQL, 2005), *Firebird* (Firebird, 2005), etc). All that eWorkbook needs, to be used with a different RDBMS, is the modification of the connection URL in the Hibernate configuration file: the creation and initialization of the DB is an automatic process.

AN EXAMPLE: THE ENGLISH KNOWLEDGE TEST

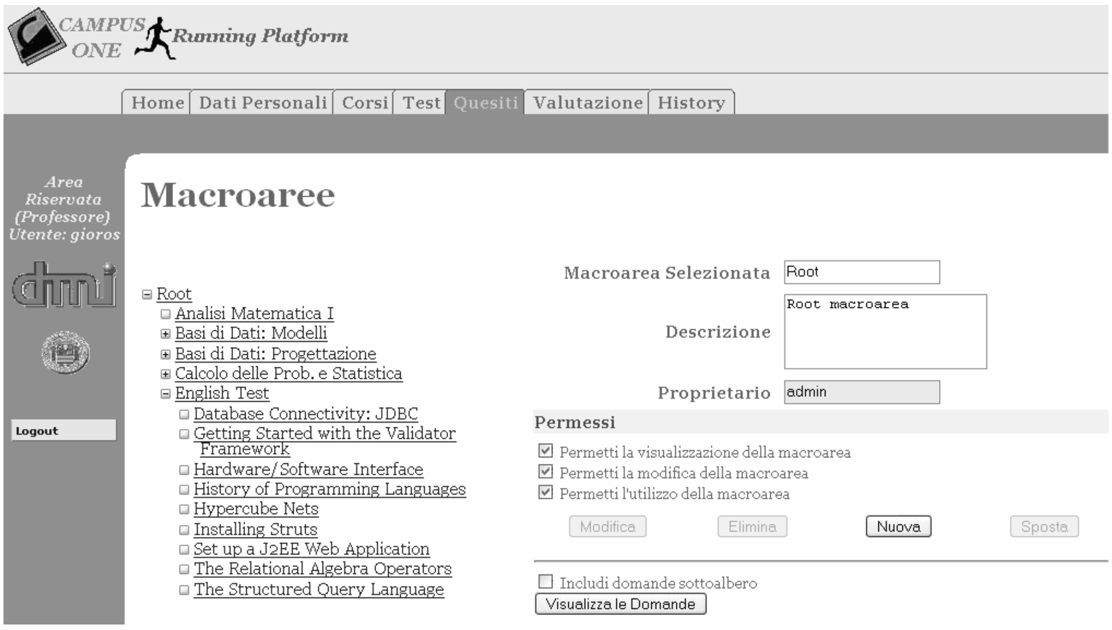
eWorkbook was installed on the Web Server of the Faculty and successfully tested for the latest sessions of the *English Knowledge Test*, which is mandatory in our university. In our faculty, the system was used to replace the traditional oral exam with an on-line objective test, more suitable for assessing a huge mass of students.

The test is aimed at evaluating learners' reading comprehension. The syllabus of the exam is composed of twenty passages taken from the textbooks of some ordinary exams. On the day of the exam each learner takes a randomly chosen passage on which his/her test is based. The time to complete the test is 20 minutes, during which the student has to answer 25 questions. A sixty-seat laboratory is available for the exams, an adequate number of users to test the system in a typical academic usage scenario.

Question and Test Authoring

In eWorkbook, the tutors can edit the question repository through a simple visual Web based interface. This is quite similar to the *my computer* browser program which allows an operating system user to edit the file system structure. As shown in Figure 4, the interface is split in two views: one on the left, which shows the question repository tree, and one on the right, which shows in an HTML form the attributes of the selected item, so that they can be easily changed. Every sub-tree on the left view can be expanded or collapsed using the '+' and '-' image controls close to the macroarea icon. A set of buttons, shown in a proper toolbar, allows the tutor to execute various tasks on the items. Each user can only browse the macroareas on which he has the UsePermission set. If an action is not allowed, the corresponding button is shown greyed.

The insertion of a question in the repository can be done through a wizard interface provided by our system. The wizard consists of a sequence of screens where the tutor must insert the question, its *stem* and *options*, the metadata and some assessment information. The insertion of a



Macroaree

- Root
  - Analisi Matematica I
  - Basi di Dati: Modelli
  - Basi di Dati: Progettazione
  - Calcolo delle Prob. e Statistica
  - English Test
    - Database Connectivity: JDBC
    - Getting Started with the Validator Framework
    - Hardware/Software Interface
    - History of Programming Languages
    - Hypercube Nets
    - Installing Struts
    - Set up a J2EE Web Application
    - The Relational Algebra Operators
    - The Structured Query Language

Macroarea Selezionata	Root
Descrizione	Root macroarea
Proprietario	admin

Permessi

- ☑ Permetti la visualizzazione della macroarea
- ☑ Permetti la modifica della macroarea
- ☑ Permetti l'utilizzo della macroarea

☐ Includi domande sottoalbero

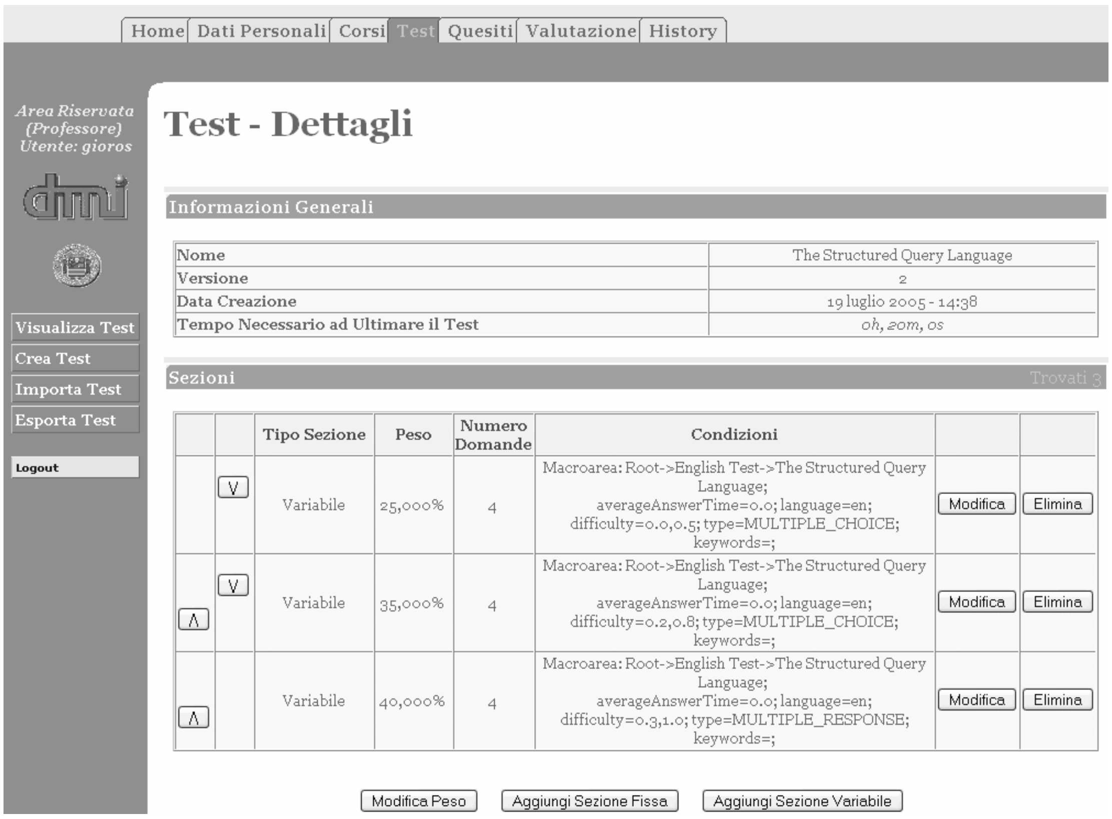
Figure 4 - A Screenshot of the Question Repository Structure

question bank is possible too: it is done by importing the question definition, from a text file or an XML text expressed in an *IMS QTI* (IMS QTI, 2005) conformant format.

A new macroarea, named *English Test*, was added to the root of the tree. A new course with the same name was activated as well. In the macroarea *English Test* twenty (one for each passage) sub-macroareas were added. In each of them, several questions were added. All the permissions for the new added macroareas and questions were set.

A new test was created for each passage. Every test is composed of three sections. The difficulty is increasing over them: an easy section containing five questions with difficulty between 0 and 0.4; a medium one containing fifteen questions with difficulty between 0.3 and 0.7; a difficult one containing five questions with difficulty between 0.6 and 1. All the tests were added to the valuable list of the *English Test* course, limiting the execution of the tests only to the computers with an IP address in the range of the laboratory in which the exam takes place.

The same test list was also published in the self-assessment section. To encourage the students to get trained, a small part of the questions used for the exam were also used for the self-assessment tests. A screenshot summarizing the test's feature is shown in Figure 5.



Test - Dettagli

Informazioni Generali

Nome	The Structured Query Language
Versione	2
Data Creazione	19 luglio 2005 - 14:38
Tempo Necessario ad Ultimare il Test	*0h, 20m, 0s*

Sezioni Trovati 3

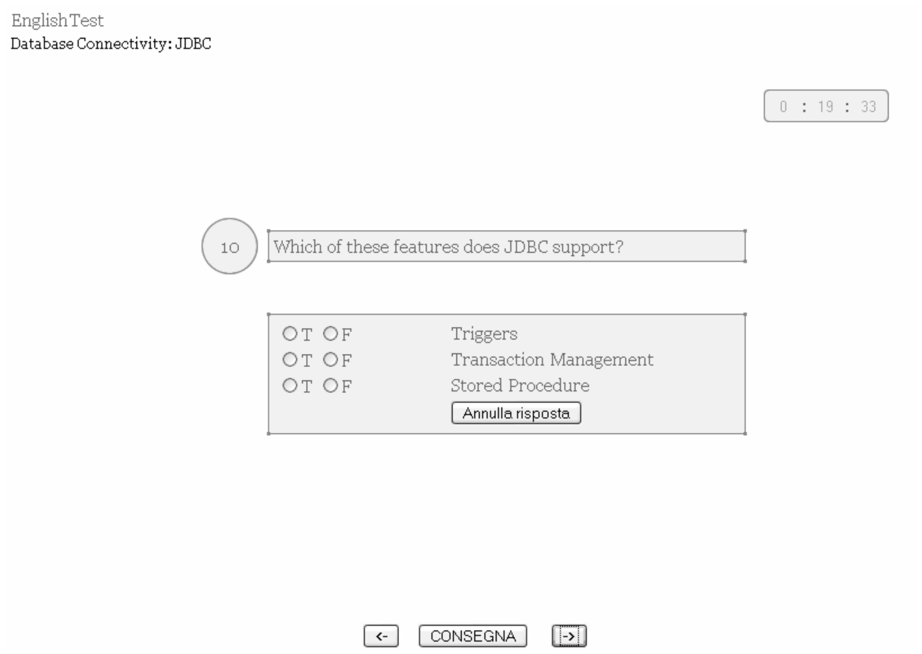
		Tipo Sezione	Peso	Numero Domande	Condizioni		
	V	Variabile	25,000%	4	Macroarea: Root->English Test->The Structured Query Language; averageAnswerTime=0.0; language=en; difficulty=0.0,0.5; type=MULTIPLE_CHOICE; keywords=;	Modifica	Elimina
Λ	V	Variabile	35,000%	4	Macroarea: Root->English Test->The Structured Query Language; averageAnswerTime=0.0; language=en; difficulty=0.2,0.8; type=MULTIPLE_CHOICE; keywords=;	Modifica	Elimina
Λ		Variabile	40,000%	4	Macroarea: Root->English Test->The Structured Query Language; averageAnswerTime=0.0; language=en; difficulty=0.3,1.0; type=MULTIPLE_RESPONSE; keywords=;	Modifica	Elimina

Figure 5 - A screenshot of the Test Details

Assessment Policy and Test Results

The *WeightedNumberCorrect* assessment strategy has been chosen to evaluate the tests: to the easy, medium and difficult sections have been given, respectively, 25%, 35% and 40% of the total score. The score has been calculated in a /30 scale, with 18 as a passing threshold. So doing, we consider a student as worthy to get the exam if he/she gets all of the easy and the medium questions and just one of the difficult ones.

All the students interested in taking the exam are asked to obtain an account on the system some days before the exam itself. Once the learner takes a test, a timer starts to measure the time he/she spends on that attempt. If he/she hasn't already done it before, he/she must deliver the test as the timer expires. Even the time spent on each question is recorded. Once the test is delivered, a table summarizing test results is shown. Two screenshots of the test execution and some pages of the test pdf format are shown, respectively, in Figure 6 and Figure 7.

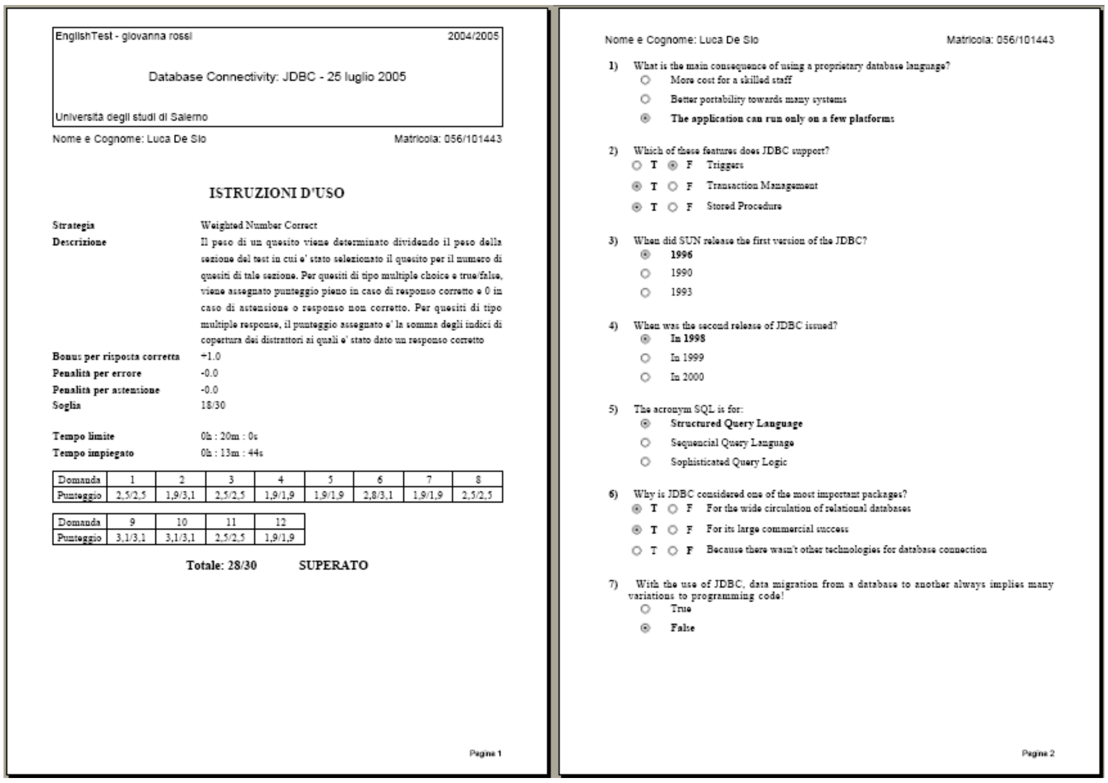

EnglishTest
Database Connectivity: JDBC

0 : 19 : 33

10 Which of these features does JDBC support?

○T ○F Triggers
○T ○F Transaction Management
○T ○F Stored Procedure

Figure 6 - A Screenshot of the Test Execution


EnglishTest - giovanna rossi 2004/2005

Database Connectivity: JDBC - 25 luglio 2005

Università degli studi di Salerno

Nome e Cognome: Luca De Sio Matricola: 056/101443

ISTRUZIONI D'USO

Strategia	Weighted Number Correct
Descrizione	Il peso di un quesito viene determinato dividendo il peso della sezione del test in cui e' stato selezionato il quesito per il numero di quesiti di tale sezione. Per quesiti di tipo multiple choice e true/false, viene assegnato punteggio pieno in caso di responso corretto e 0 in caso di astensione o responso non corretto. Per quesiti di tipo multiple response, il punteggio assegnato e' la somma degli indici di copertura dei distrattori ai quali e' stato dato un responso corretto
Bonus per risposta corretta	+1.0
Penalità per errore	-0.0
Penalità per astensione	-0.0
Soglia	18/30
Tempo limite	0h : 20m : 0s
Tempo impiegato	0h : 13m : 44s

Domanda	1	2	3	4	5	6	7	8
Punteggio	2,5/2,5	1,9/3,1	2,5/2,5	1,9/1,9	1,9/1,9	2,8/3,1	1,9/1,9	2,5/2,5

Domanda	9	10	11	12
Punteggio	3,1/3,1	3,1/3,1	2,5/2,5	1,9/1,9

Totale: 28/30 **SUPERATO**

Pagina 1

Nome e Cognome: Luca De Sio Matricola: 056/101443

1) What is the main consequence of using a proprietary database language?
- ○ More cost for a skilled staff
- ○ Better portability towards many systems
- [✓ **The application can run only on a few platforms**]

2) Which of these features does JDBC support?
- ○ T [✓ F] Triggers
- [✓ T] ○ F Transaction Management
- [✓ T] ○ F Stored Procedure

3) When did SUN release the first version of the JDBC?
- [✓ **1996**]
- ○ 1990
- ○ 1993

4) When was the second release of JDBC issued?
- [✓ **In 1998**]
- ○ In 1999
- ○ In 2000

5) The acronym SQL is for:
- [✓ **Structured Query Language**]
- ○ Sequencial Query Language
- ○ Sophisticated Query Logic

6) Why is JDBC considered one of the most important packages?
- [✓ T] ○ F For the wide circulation of relational databases
- [✓ T] ○ F For its large commercial success
- ○ T ○ F Because there wasn't other technologies for database connection

7) With the use of JDBC, data migration from a database to another always implies many variations to programming code!
- ○ True
- [✓ **False**]

Pagina 2

Figure 7 - The Test Pdf Format

At the moment, several exam sessions have been done. The mean pass rate is between 60% and 70% of the students. Some items with poor discrimination have been modified through the sessions. We finally got good discrimination on most of the questions.

Learners’ Strategies Analysis

By visually analysing the data of our experiment it came out that the most frequently adopted strategy is *Active Revising*, which was used by 40 learners over 71 (56.5%), followed by the *Passive Revising* strategy (20 learners over 71, 28.2%), and by the *Single Phase* one, used only in 9 cases over 71 (12.7%). Only two learners have adopted an atypical strategy (see Figure 2d), which cannot be classified in any of the previously described patterns.

The best results have been achieved by learners who adopted the *Passive Revising* strategy, with an average score of 17.6 exact responses over the 25 test questions. With the *Active Revising*, instead, an average score of 16.4 has been achieved. Lastly, the *Single Phase* strategy turned out to be the worse one, showing an average score of 15.1. Therefore, it appears that a winning strategy is one using more than one phase, and this is confirmed by the slightly positive linear correlation (0.14) observed between the number of phases and the score achieved on the test. For both strategies using more than one phase (active and passive revising) the score is often improved through the execution of a new phase.

The improvement is evident in the early phases and tends to be negligible in subsequent phases: starting from a value of 14.3 obtained after the first phase, the average score increases to 16.2 after the second phase. The presence of further phases brings the final score to an average value of 16.5. The average duration of the first phase in the *Passive Revising* strategy (14’50”) is longer than the one registered for the *Active Revising* strategy (11’51”). This result was predictable, since, by definition, the *Active Revising* strategy prescribes the skipping (= less reasoning) of the questions whose answer is more uncertain for the learners.

Another predictable result, due to the above arguments, is that the *Passive Revising* strategy has less phases than the *Active* one, on average (2.55 and 3.2, respectively).

The above results have been summarized in Figure 8.

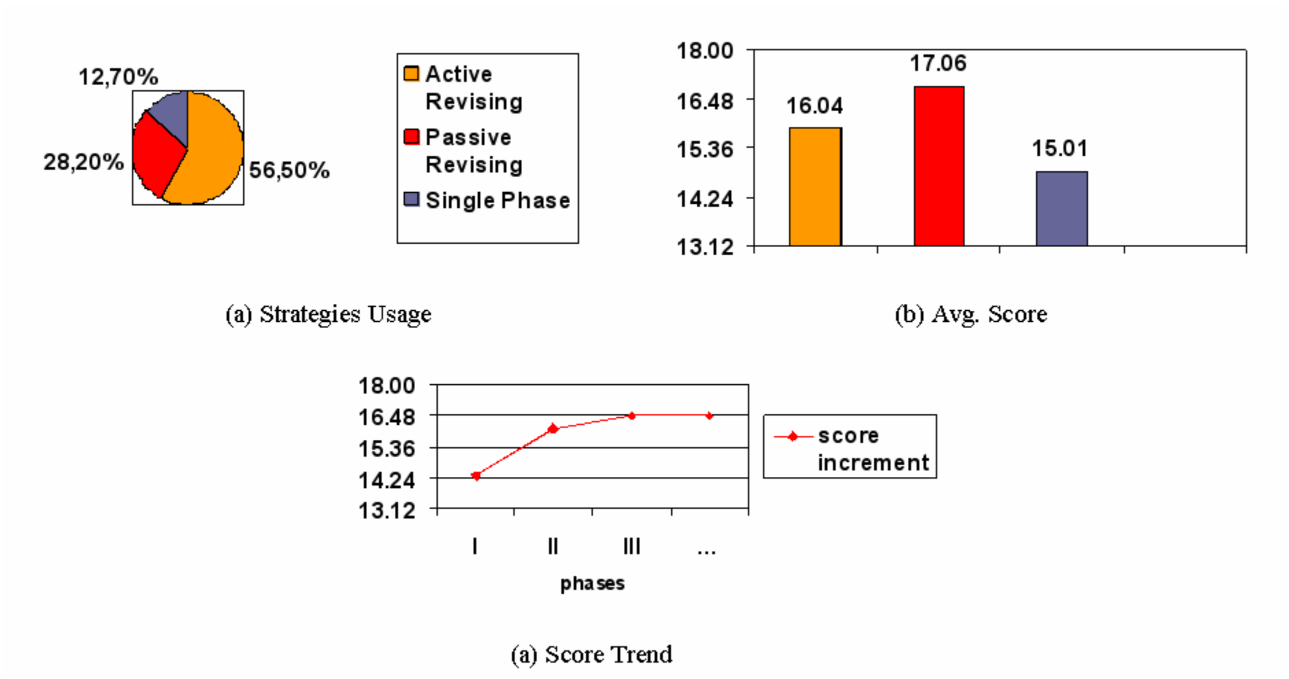


Figure 8 – Strategies Analysis

RELATED WORK

Several different assessment tools and applications to support blended learning have been analyzed, starting from the most common Web-based e-learning platforms such as *WebCT 4.1 Campus Edition* (WebCT, 2005), *Blackboard 6* (Blackboard, 2005), *Click2Learn Aspen 2.0* (Aspen, 2005), *EduSystem* (EduSystem, 2005), and *The Learning Manager 3.2* (The Learning Manager, 2005). The analysis has been carried out both by exercising the systems and by studying literature surveys and benchmark analyses (EduTools, 2005). Special emphasis has been placed on evaluating the existing systems with respect to the requirements identified in the previous section.

In the literature we can find two main categories of assessment systems: those which automatically generate questions from the lecture material, and those which make use of a pre-populated question repository from which questions are chosen randomly. The first kind of systems often requires the prior creation of a knowledge structure, like a concept graph or ontology, as for the system described in (McAlpine, 2005). Other systems of this type (Mitkov, 2003) use Natural Language Parsing to extract information from a text and generate the questions. Using these techniques, it is hard to bet on the good quality or readability of the generated questions. Such drawbacks often relegate the use of this kind of systems only to experimental purposes.

The systems which involve the tutor in the task of creating a set of questions to be stored in a repository prove to be more reliable and consequently are used more for official exams, in order to obtain an objective assessment. Those systems, such as the ones described in (Li & Sambasivam, 2003) and in (Lister & Jerram, 2001), sometimes use an XML test configuration file to define some rules for the question selection. In question repository based systems; the challenge is to give a good organization to the repository, to avoid question replication, and to use a good question selection procedure in order to assess learners' skills on the desired subjects. Some systems, like Claroline (Claroline, 2005) just use a plain container to keep questions. In Moodle (Moodle, 2005) and (Capuano et al., 2003), the question repository is partitioned in sets, often called *categories* or *macroareas*, in order to have a per-subject organization of the questions. In (McGough et al., 2001) and (Gusev & Armenski, 2002) a hierarchically structured organization of the repository is exploited. In (McGough et al., 2001), a tree is associated to a lesson and each of its branches is used for assessing learners on a part of the lesson. A leaf in this tree is a set of questions. In (Gusev & Armenski, 2002) a more complete but complex system is described, where questions are classified exploiting similarities among them.

Only a few systems adopt some kinds of author's right protection. Claroline and Moodle let the tutor choose whether to make his/her questions visible to other tutors or not.

Few systems among the analyzed ones have some forms of quality control of the questions. An interesting feature is the opportunity to judge a question or a test analyzing the learners' responses to it. Starting from this information, many criteria can be adopted. In particular, Hicks (Hicks, 2002), reporting his experience with a large class at the University of Newcastle upon Tyne, identifies a good question as the one to which the better 20% of learners answers well and the worse 20% of learners answers incorrectly. In (Lira et al., 1990) the degree of difficulty of a test is calculated using the maximum possible (max) and minimum possible (min) score and the average score (avg) of the class according to the following formula:

$$((avg - min) / (max - avg)) * 100.$$

eWorkbook has a complete tracking system to judge the quality of a question: every time a significant change is made to a question, a new version of it is generated. For each version of the question, all the history of the learner's answers is kept. From a statistical analysis, explained in detail in section *"Question Quality Improvement Through Question Lifecycle"*, we can guess the quality of the question and its improvement over time. The attempt to judge difficulty and quality of question items is not a new subject. Two main theories are noteworthy: *Item Analysis* and *Item Response Theory* (Hambleton & Swaminathan, 1985). Unfortunately, it is quite uncommon to find an assessment system that uses one of the effectively. Some explanations and a comparison between them can be found in (Fan, 1998).

As for question selection from a large database to compose tests, two algorithms were analyzed: the proposals of (Sun, 2000) and (Hwang et al., 2006).The former is aimed at constructing tests with similar difficulties. The difficulty is calculated using Item Response Theory model. The latter takes into account other parameters too, such as discrimination degree, length of the test time, number of test items and specified distribution of concept weights.

Most of the analyzed systems are complete LMS. The assessment tool is an integral part of them. eWorkbook was thought to be used by the large number of users of our university, so we gave to the didactics an organization in courses and classes, to support multiple channels in which to publish the tests.

As for a means for sequencing and control access to the tests, none of the tools analyzed has a flexible system. The system described in (Li & Sambasivam., 2003) permits the learner to sit an exam many times, until a minimum acceptable score is achieved. In (McGough et al., 2001) the questions are grouped into sets, and the strategy to pass a set, and consequently access to the next, is to give the correct response to 3 answers in a row for that set.

CONCLUSIONS AND FUTURE WORK

In the paper, we have presented eWorkbook, a system for the creation and deployment of assessment and self-assessment tests. The proposed system can significantly accelerate the assessment process, thanks to the reusability of the authored content. We achieved reusability allowing the tutors to share their questions with other tutors and adopting a hierarchical subject-based question repository. Such an organization makes it easier to find, modify and select the questions for the tests. The system is even able to interoperate with other systems that support *IMS QTI* specification. The chance to mix fixed banks of questions with randomly chosen question sections, gives the tutor the chance to get the right compromise between an objective assessment and the sureness to include a wide coverage of subjects. Author’s rights are protected through the use of separate permissions for reading, writing and using the questions.

The use of eWorkbook can help tutors in keep high the quality of the assessment, thanks to the Version Control System. This system tells the tutor if the changes he/she make to the questions positively affect the quality of the question. Other feedback information on questions are available too.

Our effort to make the application portable and usable makes it especially suitable for the academic use for which it was conceived, even though it is still a good choice in different environments. The wide choice of assessment strategies and the possibility to extend that choice with new user-defined strategies, help the tutor to tailor the test evaluation to the competency and skill level of the class. The learner can self-assess and fully reap the benefits of blended learning. The definition of access rules, like prerequisites and attempt limitation, compels the learner to follow the right learning path.

The report section is rich with information and fit out of charts and tables. The tutor can have a complete and deep control over the performance of the class and the learners even on a single macroarea, and over the effectiveness of the authored resources. The system lets tutors monitor learners' strategies during on-line tests. The approach exploits data visualization to draw the data characterizing the learner's test strategy, in order to trigger tutor's attention and to let him/her discover previously unknown behavioural patterns of learners. In this way the tutor is provided with a powerful tool that let him/her review the whole assessment process, and evaluate possible improvements.

The system has been used for the English knowledge test by the students and the teachers of our faculty. The testing has shown that teachers, also with very little technical skills, can easily use eWorkbook to create assessment tests thus fully taking advantage of blended learning. Nevertheless, a more accurate evaluation of the effectiveness of the approach is foreseen for the current academic year. Moreover, future work will be devoted to test the scalability of the system with a larger number of simultaneously on-line users. Other interesting developments are planned as future work. Although multiple choice is the most common and widely adopted question type, and it is enough to arrange structured online tests, we are working in order to support other types of questions (e.g. fill-in, matching, performance, sequencing, likert, numeric) and questions based on external tools, like those proposed in (Hicks, 2002). Other efforts will be spent to introduce multimedia elements, like images, video and sound, and rich text capabilities in the rendering of the questions.

REFERENCES

ADL (2001). *The SCORM Content Aggregation Model, Version 1.2. Advanced Distributed Learning Initiative*, http://www.adlnet.gov

Aspen (2005), *Click2Learn Aspen*, http://home.click2learn.com/en/aspen/index.asp

Bath, J.A. (1967), Answer-changing Behaviour on objective examinations. *The Journal of Educational Research*, 61, pp. 105-107.

Blackboard (2005), http://www.blackboard.com

Capuano, N., Gaeta, M., Micarelli, A., Sangineto, E. (2003). An intelligent web teacher system for learning personalisation and Semantic Web compatibility. *Proceedings of the Eleventh International PEG Conference*, St Petersburg, Russia

Chef (2005), *CHEF: CompreHensive collaborativE Framework*. http://chefproject.org

Claroline (2005), http://www.claroline.net

Costagliola G., Fuccella V., Giordano M., Polese G. (2007). A Web-Based E-Testing System Supporting Test Quality Improvement. In *Proceedings of The 6th International Conference on Web-based Learning*. Pp 272 – 279.

Dublin Core (2005), *Dublin Core Metadata Initiative*, http://dublincore.org

ECMAScript (2005), *Standard ECMA-262, ECMAScript Language Specification*, http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf

EduSystem (2005), http://www.mtsystem.hu/edusystem/en

EduTools (2005), http://www.edutools.info/course/index.jsp

Fan, X. (1998). Item Response Theory and Classical Test Theory: An Empirical Comparison of Their Item/Person Statistics. *Educational and Psychological Measurement*, 58 (3), pp. 357-381.

Firebird (2005), http://firebird.sourceforge.net/

Gusev, M., Armenski, G. (2002). onLine Learning and eTesting. *In Proceedings of 24th International Conference on Information Technology Interfaces*, Cavtat, Croatia, pp. 147-152

Hambleton, R.K., Swaminathan, H. (1985). *Item Response Theory - Principles and Applications*. Kluwer Academic Publishers Group, Netherlands

Hibernate (2005), http://www.hibernate.org

Hicks C. (2002). Delivery And Assessment Issues Involved in Very Large Group Teaching. *In Proceedings of IEE 2nd Annual Symposium on Engineering Education Professional Engineering Scenarios*, London, UK, pp. 21/1-21/4

Hwang, G.J., Lin, B.M.T., Lin T.L. (2006). An effective approach for test-sheet composition with large-scale item banks, *Computers & Education*, Vol. 46 (2), pp. 122-139

IMS ASI (2004), *IMS Global Learning Consortium, IMS Question & Test Interoperability: ASI Outcomes processing, Final Specification Version 1.2*, http://www.imsglobal.org/question/index.html

IMS QTI (2005), *IMS Global Learning Consortium: IMS Question & Test Interoperability: IMS Question & Test Interoperability Specification*, http://www.imsglobal.org/question/index.html

JSP (2005), *JavaServer Pages Technology*, http://java.sun.com/products/jsp

Johnston, J.J (1977), Exam Taking speed and grades. *Teaching of Psychology*, 4, pp. 148--149

Li, T., Sambasivam, S.E. (2003). Question Difficulty Assessment in Intelligent Tutor Systems for Computer Architecture. *Information Systems Education Journal*, Vol. 1 (51)

Lira, P., Bronfman M., Eyzaguirre J. (1990). Multitest II - A Program for the Generation, Correction and Analysis of Multiple Choice Tests. *IEEE Transactions on Education*, Vol. 33 (4), pp. 320-325

Lister, R., Jerram, P. (2001). Design for web-based on-demand multiple choice exams using XML. *In Proceedings of IEEE International Conference on Advanced Learning Technologies*, Madison, Wisconsin, USA, pp. 383-384

McAlpine, M. (2005). *Principles of Assessment. CAA Centre, University of Luton,* http://www.caacentre.ac.uk/dldocs/Bluepaper1.pdf

McClain, L. (1983). Behavior during examinations: A comparison of “A”, “C” and “F” students. *Teaching of Psychology*, 10 (2), pp. 69-71.

McGough, J., Mortensen, J., Johnson, J., Fadali, S. (2001). A Web-based Testing System with Dynamic Question Generation. *In Proceedings of 31st ASEE/IEEE Frontiers in Education Conference*, Reno, NV, USA, pp. S3C - 23-8 vol. 3

Mitkov R. (2003). Computer-Aided Generation of Multiple-Choice Tests. *Proceedings of the HLT-NAACL 2003 Workshop on Building Educational Applications Using Natural Language Processing*, pp. 17 - 22

Moodle (2005), http://www.claroline.net

MySql (2005), http://www.mysql.com

Proctiæ, J., Bojiæ, D., Tartalja, I. (2001) test: Tools for Evaluation of Learners' Tests - A Development Experience. *In Proceedings of 31st ASEE/IEEE Frontiers in Education Conference*, Reno, NV, USA, pp. F3A - 6-F3A-12 vol. 2

Sakai (2005), *Sakai Project*, http://www.sakaiproject.org

Struts (2005), *The Apache Struts Web Application Framework*, http://struts.apache.org

Sun, K. T.. (2000). An effective Item Selection Method for Educational Measurement. *In Proceedings of International Workshop on Advanced Learning Technologies*, Palmerston North, New Zealand, pp.105 - 106

The Learning Manager (2005), http://thelearningmanager.com

WebCT (2005), http://www.webct.com