A Framework for Student Knowledge Evaluation in Internet Environments

Florin Bota

Centro per i Servizi Teledidattici e Multimediali, Politecnico di Torino,

Via P. Boggio 71/A, 10138 Torino,

ITALY

E-Mail: florin.bota@polito.it

Abstract: The paper presents some considerations on knowledge evaluation in e-learning environments. The first part contains a general presentation of the necessity of automated knowledge evaluation in e-learning systems, different evaluation methods that can be used for implementation as well as advantages and drawbacks for the students and professors in using such a system. The second part presents a possible implementation, describing different original exercise types and presenting two possible implementations: self-evaluation client-side library and as a web application. The implemented exercise types are: true/false exercises, multiple choice – with multiple or single correct answers -, ordering exercises, fill-in with suggestions, fill-in with alternative correct solutions, association exercises. The client-side library has been completely implemented and is currently used for implementing different multimedia CD-ROMs with university courses. The second implementation is under development.

Keywords: e-learning, knowledge evaluation, student evaluation

Florin Bota obtained his BSc and MSc in Computer Science in 1998, respectively 1999, both from Babes-Bolyai University of Cluj-Napoca, Romania. In 2003 he obtained his PhD from Politecnico di Torino, Italy. He currently benefits of a postdoctoral scholarship from Politecnico di Torino, in the Institute for Distance Learning. His current research interests include different aspects of e-learning: knowledge evaluation, authoring tools and courseware development, use of multimedia in distance courses.

1. Introduction

On-line educational systems usually provide more functionality with respect to a traditional learning way, which uses only books as a support for student learning. Those functionalities include, without being limited to: video/audio streams, animations, simulations, different educational paths, student-student or student-professor interactivity (using mailing lists, bulletin boards and chat), remote assistance.

One of the most important functionality of an on-line learning environment is knowledge evaluation. It is one of the most difficult but extremely important parts of any educational system. In distance learning environment it can be used mainly for three purposes: self-evaluation, professor feedback and as an automated examination. The first type is allowing students to find out fast an accurate indication of their knowledge level on different topics, clearly indicating their strong and their weak points, simulating real exam situations. The students can evaluate their progress based on the feedback offered. The results obtained on this kind of self-evaluation can be available only for students or the environment can maintain them in a database and present later their evolution in time. The same results can be used as well to automatically choose a learning path from the different ones available, based on the already detected knowledge level of the student.

The second type of use is allowing professors to obtain an indication of the student knowledge. The professors can adapt the content of the course based on the results obtained by the students, in order to improve the learning environment. In case the results obtained by knowledge evaluation systems are used as an official evaluation of student performance, we have the third type of use where such systems are used instead or as a part of usual exams.

Knowledge evaluation is the only way to quantify the results in any educational system. The results can be used either by the students in order to determine the parts they have to study better or by professors in order to improve the courses content. Since the objective of any educational system is the acquirement of knowledge by the students, we can say that knowledge evaluation is an important part in any educational system.

2. Knowledge Evaluation Methods

During the time assessment methods did no change significantly 1, however most are not suitable for automatic knowledge evaluation. Most educational system use for evaluation different kinds of methods: essay papers; quiz tests, problem solving tests, projects and others. The essay papers have generally the form of a few subjects that the student has to consider in a short paper (a few pages long usually). It has the advantage of leaving the student to use his creativity, but is quite difficult to evaluate objectively and

the evaluation is generally requiring a large effort for the professor. This type of evaluation is very well fitted for human sciences (literature, languages, history and so on), but less fitted for exact sciences (mathematics, physics, computer science, chemistry). It offers little or no possibilities for automatic processing. Using artificial intelligence in some computer applications, it might be possible to automate the process in some specific area, but most cases the evaluation can be done only by a person.

Quiz tests contains a set of questions with some answers and the student has to choose the correct answers for each question. It is not allowing the students to use their creativity at all, testing only the knowledge acquired, and not the ability to use it. They have a high level of objectivity and require a low effort for the professor in evaluating the answers, but a huge effort in preparing the questions/answers pairs. Wrong answers that seem correct have to be present in order for the test to indicate correctly the knowledge level of the students, and preparing those "fake" good answers require a good knowledge of the usual mistakes made by the students. Quiz tests are well fitted for most of the fields, so they can be used with good results in most of the educational systems. This type of evaluation can be automated very well, requiring the professor to place the questions/answers pairs in some databases, while an application would check the student answers against the database.

Problem solving tests contain a few problems that the student has to solve, using for that the knowledge he acquired. It is one of the most difficult tests for the students, since it requires not only knowledge, but also understanding and the ability to use the knowledge. Their level of objectivity vary, usually being a medium one, due to the fact that usually the problems can have multiple solutions and most of the students do not finish the presentation of the solution, so that the professor can not evaluate their answers based on a final result, or by checking to see if the presented solution really solves the problem. The professor would usually try to understand the ideas that lead the student to choose that specific solution. Problem solving tests are well fitted for exact sciences, where the students have to develop the ability to use the knowledge, not only to acquire it. This type of evaluation is very difficult to process automatically, even if it would be quite simple in some cases to implement an application that would automatically check whether or not the solution proposed by the student solves the problem. In the case that the solution is only partial, or is not a correct one, an automated evaluation would probably be impossible.

Other types of evaluations try to evaluate the ability of the student to solve a bigger problem, working in a team (project work), speech (oral examination), but are not fitted at all for automated processing.

There are two main possible implementations of automatic knowledge evaluation systems: desktop based and client-server. The first one can be used in computer based training (CBT) for self-evaluation and different internal purposes – such as content adaptation, automatic educational path selection or others – while the second one can be employed as well for professor feedback and on-line examinations.

Initial implementations generated paper sheets containing quiz exercises for examinations and scanned the sheets with student answers for evaluation. This allowed faster and more objective results; later implementations offer support for creating tests based on different criteria, as shown in 2 and on-line evaluation.

Different educational institutions introduced on-line implementations for student examinations, using them mostly in experimental way. Typical automatic knowledge evaluation implementations use only quiz exercises, supporting eventually multiple correct answers or "partial correct" answers that correspond to lower grades. Other methods are possible, for example, in 3 is presented a method for evaluating student performance using web-server logs for this purpose.

C. Cárdenas presents in 4 a model used for academic evaluation in a campus-wide implementation, where an on-line system offers an accurate evaluation of student performances before a final, nation-wide, graduation exam.

3. On-line assessment systems

Implementing assessment systems as on-line applications for students has different advantages and disadvantages, both for students and teachers, indicated by Dottie Natal in 1.

Advantages for students are:

- Possibility for "any time, any place" testing;
- Support for student with disabilities;
- Valuable class-time is not used for exams, but for study instead;
- Immediate feedback may be available;
- Access to tools they are comfortable with.

Some of the disadvantages for students are:

- Procrastination – last minute test taking;
- Availability of computers;
- Lack of computer skills;
- Inability to use their standard test-taking skills (answering the hardest or sure answers first).

Some of the benefits for teachers are:

- Decreased record-keeping time;
- Reusable test questions;
- Higher legibility than for hand-written tests;
- Increased teaching time;
- Better test-design – for different learners;
- Possibility to have more frequent assessments;
- Better statistical feedback.

Possible drawbacks for teachers include:

- Security problems;
- Unexpected results with regards to writing standards;
- Debugging problems with feedback or test administration;
- Hardware problems.

Two different scenarios can be used for implementing an on-line examination 5: a dedicated laboratory or over the Internet. The former one allows students to sustain the exam from a specific place, typically a computer laboratory, with a supervisor monitoring the activities, in which case it is possible to have students not able to sustain one exam due to unavailable computers. The later allows students to use any Internet connected computer for taking the exams, however imposes different other problems: are the students cheating, by using books, documents, web documentation? Who is really taking the test, is it possible to verify that actually the student is the one in front of the computer and answering the questions? In most cases the first scenario is used for examination, where the results are used for official records, while the second one can be successfully used for self-evaluation or automatic course content adaptation, where the teacher does not care whether or not the student is cheating, as long as he is learning something.

Different methods can be used to measure the student performance for multiple choice computer based tests 6:

a) Good/bad question answer;
b) Wrongness searching;
c) Correctness searching;
d) Combined model;
e) Statistical model.

The *good/bad* model considers that each question is answered either correctly, either wrong. The system would simply count the correctly answered questions and the final mark would be calculated based on this count and the total number of questions. This model has the big disadvantage that it does not allow for a very good evaluation, considering the fact that a student might answer partially to one question and still get no credit for that.

The *wrongness searching* model grants in the beginning all the points (the maximum mark) and penalize the student for each wrong answer. This is not a very good model for an automated evaluation; since a student that would not answer at all would probably get the maximum mark (he surely did no mistakes, since he didn't answer anything).

The *correctness searching* model grants a minimum mark in the beginning and awards a certain amount of points for each correct answer. For exercises with multiple choices this is not a very good model for an automated evaluation, since a student that would check all the answers would probably get the maximum mark (he checked everything, so all the correct answers were checked). The problem can eventually be solved by limiting the number of choices a student can check.

The *combined model* grants in the beginning the minimum mark, awarding some points for each correct answer and penalizing for each wrong answer. This is a better model for an automated evaluation, not allowing those not answering at all or checking everything to get the highest marks.

The *statistical model* is a variation of the combined model, where the proportion between the points awarded for a good answer and the penalty points for a wrong answer are calculated based on the proportion between the number of correct and wrong answers in the test. This way it is possible to have both marginal cases (checking everything, checking nothing) getting the lowest mark.

4. Evaluation Framework

In 1999 a basic evaluation system using multiple choice questions has been implemented and is currently used at the Faculty of European Studies, "Babes-Bolyai" University for the theoretical part of computer-science exams 6. At Politecnico di Torino a variation of the system has been used for automatically detecting the knowledge level of a student and adapt course content to his needs, in XML Internet learning environment 78. In those implementations has been used a statistical model for assessing student answers, where the proportion between the awarded points or penalty points was calculated based on the proportion of correct/wrong answers for each question.

Based on this work a complete knowledge evaluation framework is currently under implementation. While the basic one supported only multiple choice questions, the framework offers different other types of exercises: true/false exercises, multiple choice with one or multiple correct answers, associating exercises, ordering exercises, fill-in and others. Two different versions have been considered: a self-evaluation client-side library or a client-server web application. The former is implemented in JavaScript and can be used on simple web sites and multimedia CD-ROMs for student self-evaluation, since the functionality does not depend on server-side programs. The later will be based on the use of web programming and will maintain data on SQL servers, implementing a knowledge evaluation environment.

Both versions use JavaScript and DHTML for generating the content visualized by students based on certain parameters. The HTML code of the web page with the exercises is very similar, since the client-server implementation can use a small variation of the visualization part of the client-library for dynamically generating the visualized content. A web page generated from the client-server implementation can be easily modified in order to be used on simple sites or CD-ROMs for self-evaluation. It would be quite simple to implement a small program that would generate from the data maintained on the SQL server the complete web pages needed for the self-evaluation versions.

The client-side library has been completely implemented for Internet Explorer 5.5 and above, support for Netscape is currently under development. For the client-server version has been implemented the part for the visualization of the content and evaluation of student answers, while different other functionalities are still under development.

5. Exercise Types

Each exercise has a graphical aspect containing a small description. Both implementations support the same generic exercises, that can be used in different courses and which are presented in below.

5.1. True-false exercises

This is one of the simplest exercises where the student has to indicate for each one of the affirmations contained in a list whether or not he considers it true. Due to the graphical implementation of the visualization of "true" and "false" values it is possible to use variations where they are replaced by "yes"/"no", "correct"/"wrong", "activated"/"deactivated", "open"/"close", practically whatever pair of values with Boolean interpretation (see fig. 1). For each affirmation the author can indicate the number of points to be granted in case of a correct answer, as well as two different explanative messages for the user in case of a correct, respectively wrong answer. The user chooses the answer he considers correct and the system will grant him the number of points indicated by the author in case the choice was correct or penalizes him for a wrong answer. The system should deny the selection of both possible answers for a single affirmation – when the user selects an answer, automatically the other one is unselected –. The solution presented afterwards indicates the correct choices (see fig. 1), and in a separate window explains the user the mistakes and shows him the points obtained.

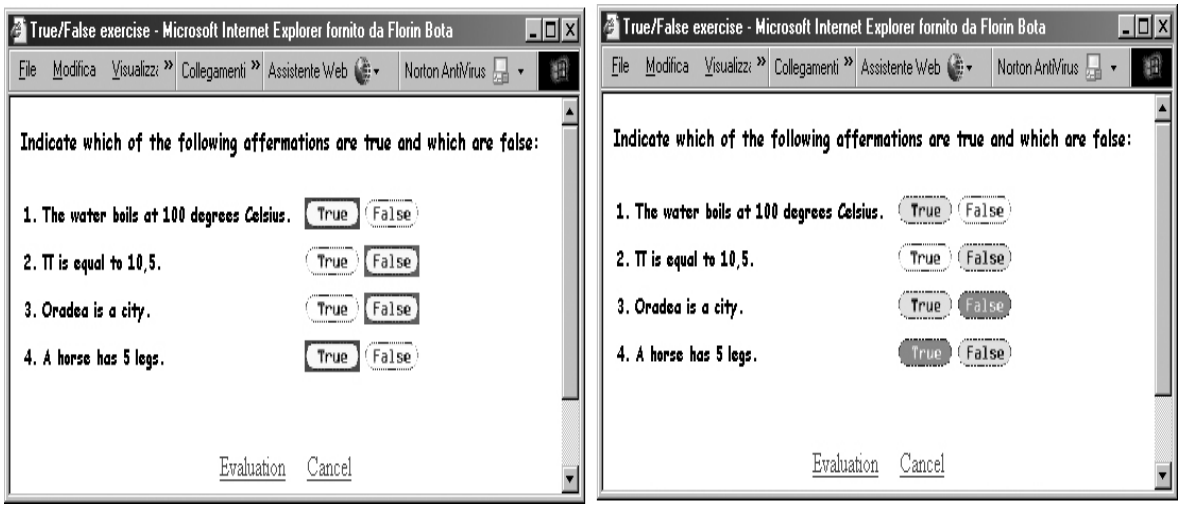

Figure 1: Visualization and Solution of a True/False Exercise.

5.2. Multiple choice exercises

This is a well-known type of exercises, used for automatic knowledge evaluation. The student is presented with a small text and a list of possible answers, he has to choose the ones that are correct. Two different cases have been identified: a single answer is correct, respectively multiple answers can be correct. In the first case, the system can automatically unselect all other answers when the user selects an answer, while in the second case this is not possible. The difference is mainly due to the allocation of points: for the first case is not obligatory to implement a penalizing system. For the second one it is necessary either to penalize the student for wrong answers or to implement a system blocking the selection of more choices than the number of correct answers this becomes necessary due to the possibility to select all answers. In the client-side library version has been chosen to limit the number of answers that the user can select, in order to eliminate the possibility to select all of them. This way it is not mandatory to implement a penalizing system, although one can be used by author if assigning a negative number of points to some or all wrong answers. In fig. 2 is shown the visualization and the solution of one such exercise.

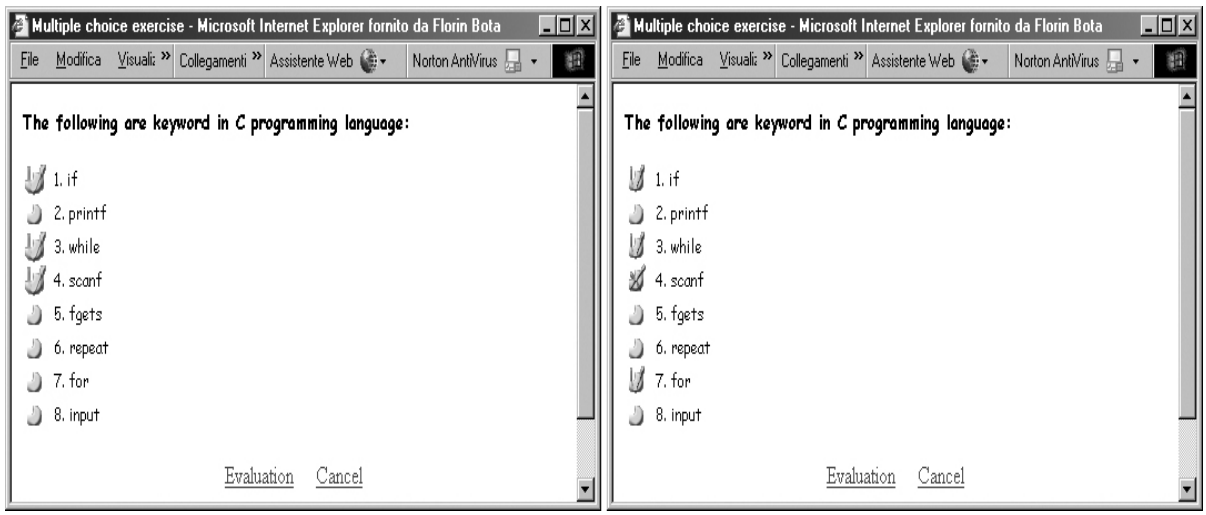

Figure 2: Multiple Choice Exercise Visualization and Solution.

5.3. Ordering exercises

For this type of exercises the user is presented with a list of items and has to indicate the correct order for them. For example this type can be used for ordering the different phases of a technological process, or use ask the student to use specific criteria in ordering the items (alphabetically, numerical order, chronological etc.). The author indicates for each item the correct position, the number of points that should be assigned in case the user correctly indicates the order and explanatory messages. In the client-library a fill-in space is presented at one side of each item, where the user has to compile the desired order (see fig. 3). The system is checking whether the user indicated a correct order for all items, without repeating the same one for two or more items. This assures that

mistypes do not lead to incorrect answers. The system grants the user the points for each item for whom he indicated the correct position. The solution (see fig. 3) indicates the correct ordering of the items together with user-proposed order and eventually explanations.

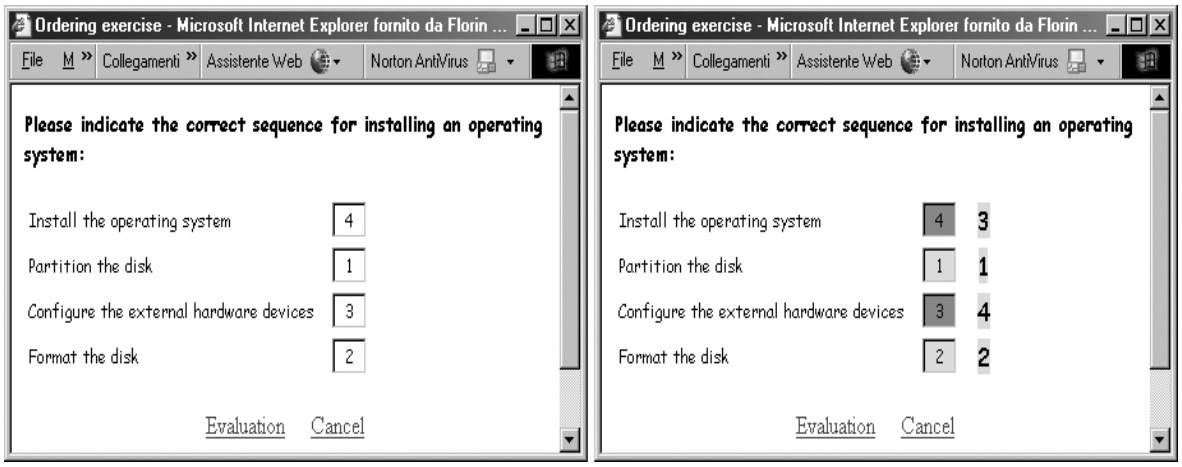

Figure 3: Ordering Exercise Visualization and Solution.

5.4. Fill-in with suggestions exercises

This is a fill-in the blanks exercise type, where some possible solutions are suggested to the user. Practically, a text where some parts are missing is presented to the user, which has to fill in the missing parts. For each missing part apart of the solution, the author can indicate up to five "fake" solutions, all presented to the user in a random order at the end of the text (see fig. 4) as possible solutions. This type of exercise is very often used in grammar or foreign languages knowledge evaluation, but can be applied for other fields as well. The text the user proposed can be checked in different ways – case sensitive/insensitive, with or without white space elimination – in order to allow different degree of difficulty. For each missing part that the student fills in correctly, the system grants him the number of points indicated by the author of the exercise.

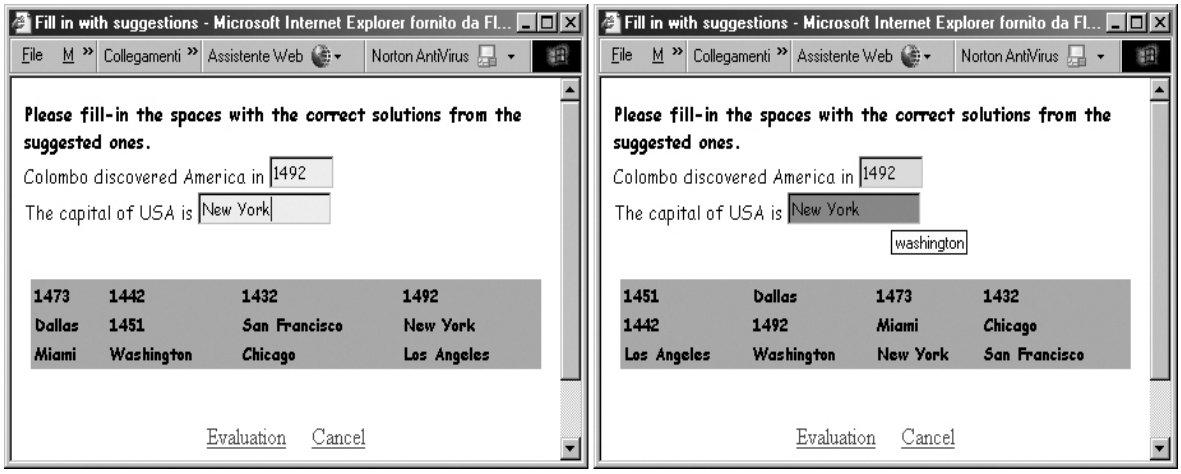

Figure 4: Fill-in with Suggestions Exercise Visualization and Solution.

5.5. Fill-in exercises with alternative solutions

This exercise is similar to the previous one; the author can indicate up to five alternative solutions, beside the main solution – the one that is considered to fit best in the context –. All solutions are considered correct by the library. The user is presented only with the text and has to fill-in the missing parts. As in the previous type, the check can be done case sensitive/insensitive, with or without white space eliminations. The solution proposed by the student is compared to all alternative correct solutions, and in case a match is found the student is granted the number of points specified by the author of the exercise. The visualization of the result, as for the previous exercise, indicates the main solution and the parts the student solved correctly.

5.6. Association exercises

For this type of exercises the user has to make associations between different items. The author of the exercise can choose between a simpler version, with two sets of items to be associated, and a more difficult one, with three sets of items. The author has to indicate the correct association, as well as the number of points to be granted to the user for a correct association – for the three-sets version a partial association will lead to the system granting a part of the points to the student –. The student is presented with one column for each set of items and has to indicate the associations he considers correct (see figure *5*). In this implementation has been implemented a limitation of having an association of one-to-one, however more complex versions could be implemented, where one-to-many associations could be used. This restriction simplified the interface, since a simpler one where each item identified by a system generated code: arabic numbers (1, 2, 3 and so on) for the first column; letters (A, B, C and so on) for the second and double arabic numbers (11, 22, 33 and so on) for the third column. Up to nine different items can be used for a set in case of three sets of items, while in case of only two sets up to twenty-six different items can be used in a set. The user can indicate in specific fill-in spaces the desired associations. In case a one-to-many association a different interface for obtaining user choices should be used.

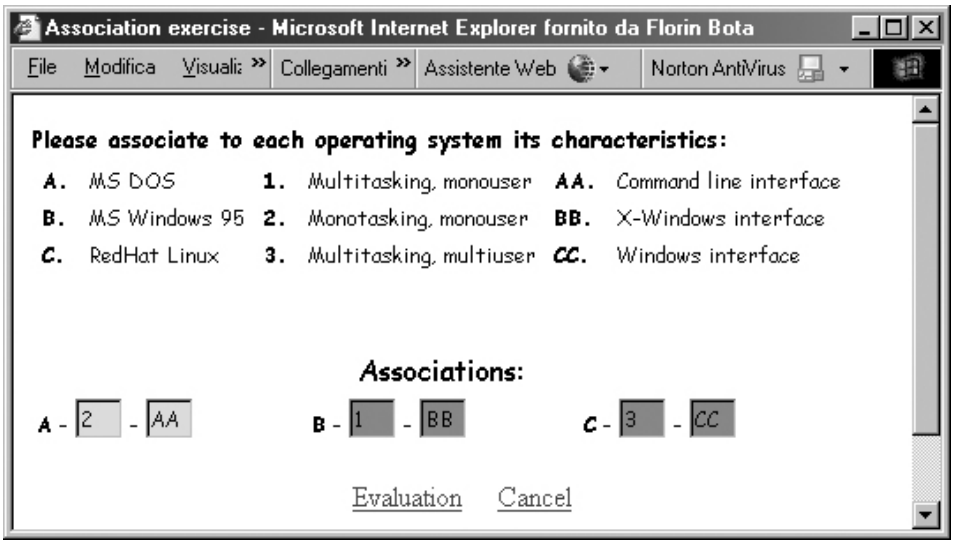

Figure 5: Solution of an Association Exercise.

6. Implementation Details

The client-side version implements an object-oriented JavaScript library, allowing for the creation of exercises and possible answers. The constructor of the exercise receives different parameters, indicating the exercise type, number of possible evaluations the visible text and an array of possible answers. For the creation of a possible answer, the author has to indicate apart the visible text – which can be HTML content – some other parameters: explanations and different values indicating the correct solutions, suggestions, depending on the type of exercise. The library contains a function that dynamically generates the visualized content.

The implemented Java-Script library uses cookies to maintain different information and offer reports to the student. Each exercise is identified by a unique code and is part of a group of exercises, named section. The author of the exercise can indicate the maximum number of evaluations the student can effectuate, before the system has to consider as final the proposed solution. Different suggestions can be indicated by the author, which are presented to the student after each evaluation in case the solution was not correct. After the last evaluation the student can see the correct solution as well as explanations of the mistakes he made. The author can indicate two separate explanations: one in the case the student offered a correct solution and a different one for an incorrect one.

The library saves the codes of the exercises the student already solved into a cookie. While dynamically generating the visualized exercise content, the library checks the values saved into the cookie and in case the student already solved the exercise he is no longer allowed to solve it. The library maintains as well the number of exercises the student solved, respectively the number of correct proposed solutions and number of points obtained for each exercise. At the end of each section a page presents a report to the user on his performances, indicating the final grade and information on the solved exercises. The same report page offers the possibility to clear all content maintained in the cookie, in order to restart the self-evaluation.

Due to restrictions imposed on the size of the cookies by browsers, it is not possible to maintain user responses in order to offer a final review of the mistakes he made.

Beside the functionalities of the client-side library, the implementation as a web application will have some new functionalities, mainly due to added possibility for data storage and processing on server side. Exercises and different answers are proposed randomly from a set of possible exercises existent in the database, so that two students will not get the same exercises. All student answers will be saved to the database allowing students to interrupt and resume later their test, as well as review a specific test taken some time before. This offers as well the possibility to use a strategy more similar to classical test, answering sure or easier questions first, since the application is automatically saving student answers. A time limit has been implemented in order to limit the total amount of time the student can pass solving an individual exercise or the whole test. In order to have a better granularity of the knowledge a clustering of the exercises will be used, grouping the exercises in function of difficulty levels. In this case based on user knowledge detected from previous exercises it will be possible for the server part to propose new exercises with a degree of difficulty adapted to the user, so that users with a lower knowledge level will get easier questions – obviously with lower points for correct solutions – while students with higher level of knowledge will get more difficult exercises – with specific bonuses for correct solutions –.

7. Conclusions

The paper presented a possible implementation of a knowledge evaluation system, using different new types of exercises apart from the classical multiple choice ones. The client-side library is already in use for implementation of multimedia courses on CD-ROMs at Politecnico di Torino, while the web implementation is under development. The web implementation will be used mainly for student self-evaluation, but it is expected that some professors will consider replacing whole or part of the final exam for their courses with this application. The blocking of the exercises ensures the user will not resolve the exercise in order to obtain a better grade, while the reporting functionality offers valuable information to the user regarding his strong and weaker points.

REFERENCES

1. NATAL, D., On-line Assessment: What, Why, How, **Technology Education Conference**, Santa Clara, California, 1998, pp. 1-23
2. J. PROTIC, D. BOJIC, I. TARTALJA, test: Tools for evaluation of students' tests – **A development experience**, Frontiers in Education Conference, 2001. FIE 2001, Reno, USA, 2001, pp. F3A-6-F3A-12 vol.2
3. M. RAHKILA, M. KARJALAINEN, **Evaluation of learning in computer based education using log systems**, Frontiers in Education Conference, 1999. FIE '99. 29th Annual, San Juan, Puerto Rico, 1999, pp. 12A3/16-12A3/21 vol.1
4. C. CÁRDENAS, **Development and implementation of a new model for academic evaluation**, Frontiers in Education Conference, 2000. FIE 2000. 30th Annual, Kansas City, USA, 2000, pp. F3A/15-F3A/20 vol.2
5. M.J. GRANGER, N. MCGARRY, **Incorporating on-line testing into face-to-face traditional information systems courses**, 17th Annual Conference of the International Academy for Information Management, Barcelona, Spain, 2002, pp. 220-226
6. F. BOTA, L. FARINETTI, F. CORNO – **Student Knowledge Evaluation in Internet Environments**, paper presented at the Fourth Romanian Internet Learning Workshop - "Internet as a Vehicle for Teaching", 26 June – 8 July 2000, Sumuleu-Ciuc, Romania
7. F. BOTA, L. FARINETTI, A. RARAU – **An Educational-Oriented Framework for Building On-Line Courses Using XML**, IEEE International Conference on Multimedia and Expo, 2000, pp. 19 -22 vol.1
8. L. FARINETTI, F. BOTA, A. RARAU – **An authoring tool for building flexible on-line courses using XML**, proceedings of "XML Europe 2000" conference, Paris, France, 12-16 June 2000, pp. 77-81.