Abstract: The introduction in 2001 of a centralized E-learning platform at K.U.Leuven brought challenging opportunities for accounting courses to engage in E-learning. This paper deals with the problems encountered in the development of electronic tests and remediation exercises. First a list of requirements, to make the exercises as realistic as possible, is presented. Then we give an overview of issues with the available ICT platform and we end with a discussion of the newly developed software.

Introduction

In 2001 K.U.Leuven chose a centralized approach (Toledo 2002) for implementing ICT to support courses and learning activities in one learning platform based on two commercial products. For electronic courses K.U.Leuven opted for BlackBoard and for electronic test and remediation exercises Question Mark Perception1 (QM 2002) was implemented. For most of the courses at the Faculty of Economic and Applied Economic Sciences, going from paper to other media only caused minor problems. Making electronic versions of accounting exercises related to accounting courses is a lot trickier and the focus of this paper is to elaborate on how electronic test and remediation exercises were implemented (GIZEH 2002). The main problem to tackle has been the typical tabular format for accounting. Not finding any satisfying commercial product resulted in building an interface between QM and a newly developed tool. The goal was to enable students to do their tests and exercises in the same flexible way as they would be doing them on paper. The reason ICT was preferred to paper boils down to time management. Using ICT for these tests and exercises allows providing students with fast evaluation and feedback because it is generated electronically. Tutors can now focus on providing support to students really needing it without losing time with batch correction jobs.

Requirements

First of all the exercise’s representation should look as natural as possible. This limits the possibilities of multiple-choice questions to assessing the theoretic parts of the course. For exercises a representation close to the T-like scronto format or the tabular journal format is needed as these are the two formats allowed according to the Belgian’s accounting standards. Another requirement was that the tool should be very flexible when dealing with inaccuracies of the student. This was realized by providing ranges for account numbers and ranges for amounts. The ranges of account numbers were to overcome the situation were one or more account number in a group is considered correct. To bypass rounding errors during calculation the ranges for amounts were provided.

Finally specific requirements to aim at for scoring questions and providing feedback were outlined. The answers will be evaluated based on two factors both worth half the exercise’s score. The first part considers the correct structure, meaning the correct account numbers and corresponding account descriptions, which is either correct or incorrect resulting in null or the full marks. If the structure is correct the second part is evaluated were the accounts’ amounts are looked at. Every correct account is worth the same and the score is expressed as percentage to the maximum given by the correct answer making up the score for the second part. The feedback is adapted according to the student’s answer. If the student returned a blank answer he will only receive a hint indicating where in the course the answer can be found what discourages students trying to collect exercises.

[1] the version of QM referenced to in this paper is 2.5.0.0
When the structure was answered incorrectly the student receives the correct structure as feedback and the feedback for mistakes against the accounts’ amounts is the correct answer to the exercise. Every feedback should also provide a unique reference label. This label enables students to refer to specific exercises during communications with their tutors.

**Challenges & Solutions**

In general we found out that accounting exercises do not fit well in the features provided by E-learning ICT tools because of the lack of flexibility needed as indicated in the requirements. Initially we tried to use basic question types available in every ICT product. While multiple-choice question could be used in theoretical questions they were ruled out very quickly as accounting has nothing to do with selecting the correct answer. A second attempt was made with fill-in-the-blanks questions. Here we were able to get very close to the requirements but some semantic and syntactical problems also ruled this out in the end. Other question types were not considered useful.

Because it is used principally in exercises, the tabular journal format was implemented (Figure 3). It represents the different participating accounts in one accounting transaction. In the first column the account numbers and account descriptions are shown grouped together around the ones that are debited, before the ‘@’-symbol, and the ones that are credited. In the following columns the amounts by which the corresponding accounts are debited, first column, or are credited, second column, are indicated. The first property laid down was the size of the table. It should allow the students to answer most, if not all, types of questions. We opted for one fixed format with four debit accounts and four credit accounts. Providing exactly as much rows as needed would give away the answer as students will start recognizing answer patterns and providing the number of row in a dynamic way has very high load times and evaluation response time because of the dynamic structure’s size and complexity. The solution with the fixed table size overcomes both problems and is still very flexible. The students are allowed to enter their solution in any order, as long they respect the debit and the credit groups, irrespective of the position of unused blank space.

The things we could not do flexible enough in QM using its basic features lead to the implementation of an interface with QM. Firstly, we would like to complete the account description when the student enters an account number as every account number has a fixed description. For a student this is valuable feedback. Secondly, we like the required flexibility dealing with the way a student enters his solution. As indicated before the order and the position of accounts and blank unused lines should not play any part. Semantic problems made it impractical to implement using these basic QM features.

The semantic problems have to do with the way exercises are corrected and the way the maximum score is calculated. For correcting exercises a set of logical expressions is used. In QM logical expressions may not contain parentheses to guide the evaluation and consequently have to be serialized to overcome this. However due to combinatory explosion during the serialization process, because of the nature of accounting questions, the expressions surpassed the question size upper limit. A second semantic problem is about the way QM calculates the maximum score of a question based on a set of logical expressions. QM calculates the maximum score by adding all the scores potentially given by every expression in the set together not taking into account that some expressions might be mutually exclusive. It is possible to overcome this by intervening directly in the DB, storing the QM generated exercises, and by adapting the scripts that generate the grading reports. Since this is dependent on the QM version it was not considered an option.

QM is not optimal for representing accounting exercises directly but still used, be it through an interface, and there are some good reasons for this. First of all QM is a standard with K.U.Leuven so support is guaranteed. Second it is better for students to have a limited set of standard tools they have to work with. The last argument is that if we implement an interface with QM it is possible to use its report generator, deployment options and user management without having to implement this ourselves.
Implemented Solution

The relation between QM and the developed software is indicated in Fig. 1. The software basically consists of two parts: an authoring tool, B2QML, used by the tutors to generate the accounting exercises and a Java applet, QMB, used by the students to do their exercises. The software satisfies all the requirements previously indicated.

![Figure 1: Relationship between QM, B2QML and QMB](image)

B2QML (screenshot in Fig. 2) is capable to export exercises in the QML import format for QM questions. B2QML checks the validity of the question and verifies the question for semantic author mistakes. Generating accounting questions directly in QM with all the evaluation aspects is very complex what makes it error prone and therefore we chose to use an authoring tool instead. Moreover HTML is used in the deployment of questions, QMB applet encrypted parameters have to be set and the logical expression for the evaluation have to be written down which was considered not to be within the typical author’s range of skills.

![Figure 2: Screenshot B2QML](image)

The student interacts with the QMB applet (screenshot in Fig. 3) that is embedded in a dynamic web page generated by QM. QM passes, through parameter binding, all the necessary information to the applet to enable the applet to evaluate a specific exercise. When the student submits his answer the QMB applet passes the score back to QM and the latter will generates a web page containing the feedback according to the score. Because the answer to the exercise is visible in the source of the generated web page as the parameter list of the QMB applet we apply an encryption algorithm to the parameters. The applet provides the author with an option to reveal some aspects of the answer. These parts are shown as pre-filled content fixed input zones in the QMB applet.
There are some other advantages to this approach. The QMB applet can be embedded in any web based environment as it is a stand alone piece of software. In the near future it will be used in a CD-Rom based accounting self study package implemented by means of Praktix Presentation Viewer (PPViewer 2002) which is a table based hypermedia viewer developed at K.U.Leuven. The authoring tool, B2QML, is language independent. The whole tool is based on definition tables. Only the translation of these tables is needed to implement a version of the tool in another language what makes the tool very adaptable and allows for easy maintenance.

Future Development

For the moment the solution discussed in this paper only allows accounting exercises represented as one entry transactions in the journal. Viewing them as a series of one-entry transactions and providing a QM question for each of them is the only way to support exercises having more entries. Implementing a real multi-entry exercise would make the exercises even more realistic compared to real live accounting but the trouble is the combinational explosion during evaluation of such exercises. Belgian accounting standards do not provide a fixed order and moreover it allows entries to be combined or uncombined leaving the implementation as a very complex but tempting problem for future development.

References


