Bsc Project

Office Skill Assessment

Final report

Delft University of Technology

Egbert Bouman 1099639

Date: December 11, 2008

Preface

The final part of the bachelor of my study Technical Informatics at the Delft University of Technology consists out of an internship. Usually this internship is performed outside of the university. Unfortunately because I am suffering from CANS I am not able to perform this internship full-time. Therefore I did the internship part-time for CICAT. CICAT is the central liaison office at TU Delft, it provides management support in the coordination of international projects.

The typical duration of an internship performed as part of the bachelor project is twelve weeks. Because I did the project part-time, the duration of the project was extended to last until December, making the total time for the project close to 9 months. I started working on the project on March 9th 2008, and I finished the project December 11th 2008.

During this time I worked on a part of a larger system used to assess the Office skills of students. The main subject of my part in this project was integrating the system into Moodle.

First and foremost I would like to thank ir. Bernard Sodoyer of the Delft University of Technology for giving me invaluable help and advice during my work. Also I would like to thank him for giving me the opportunity to do this internship part-time. Second, I would like to Bert Geers from CICAT for giving me the opportunity to be involved in this project. And finally I would like to thank my colleague's for the pleasant environment in which we worked.

Delft, December 2008
Egbert Bouman

Table of contents

Summary

The universities in Sri Lanka currently do not have the required resources to meet the demand for education. Therefore only the best students are allowed to study at a university. To make studying available to larger groups of students a new Bachelor of Information program has been setup at the University of Colombo School of Computing (UCSC). This new program will be completely based on e-learning. Part of the program will be a course that is concerned with the Office skills of students. In 2007, during an internship at the UCSC in Shri Lanka, a system prototype was created that is able to assess certain Word skills of students. The prototype however has never been used, because it needed to be extended and improved. This year, the Office-skill assessment project has been continued to make these improvements.

The UCSC currently uses Moodle, an open-source course management system, to help educators create e-learning courses. The Office skill tests have been integrated into Moodle so that no separate system is needed to allow students to take Office skill tests. Also, the UCSC requested some customizations to the Moodle code to make it more usable. The specific part of the project that this document is about concerns Moodle integration and customization.

In Moodle educators have the ability to add quizzes courses. When a student is enrolled in such a course, he or she can take part in the quiz. The functionality of the quiz module can be extended to include various types of questions by the use of question type plugins. Therefore, a question type plugin has been made to include Office skill questions. The officeskill_qtype class is one of the most important classes of the plugin. For example, it pushes grades to the gradebook and handles user responses. It does this by extending the functions of the default_questiontype class.

As for the customizations, the quiz edit page has been altered to include a checkbox indicating whether a user wants to select a random subcategory. When a number of random questions are added to a quiz with the checkbox set, this group of questions will all be selected from the same random subcategory.

The random question type has also been altered to support question conditions. The management of these conditions takes place using the "Question conditions" page of the quiz edit tab.

Furthermore, a customization was made to include the amount of seconds of extra time is given to a certain attempt at a quiz. This extra time is then added to the timelimit for the quiz. The extra time can be set from the overview page of the quiz results tab.

The final customization that was requested was to make Moodle capable of recovering a damaged cookie of the client. It is not possible for Moodle to restore a cookie, so the next best thing would be to disable cookies altogether. With some adjustments, Moodle can be forced to use cookieless sessions. When this option is active the session ID (used to track the user session) is made a part of the URL. Most browsers remember the URL that was visited, so that when the browser is restarted, the same page is loaded again. However with this design, there are also some security considerations. Because of this, the idea has been discarded. Restoring a cookie should be done by the client's browser instead.

The project can be considered a success. The Office skill system is now fully integrated into Moodle. The customizations for Moodle were also successfully completed, with the exception of ability to recover the cookies used by Moodle.

1. Introduction

1.1 Office skill assessment

The universities in Sri Lanka currently do not have the required resources to meet the demand for education. Therefore only the best students are allowed to study at a university. To make studying available to larger groups of students a new Bachelor of Information program has been setup at the University of Colombo School of Computing (UCSC). This new program will be completely based on e-learning. Part of the program will be a course that is concerned with the Office skills of the students. There are only 3-4 teachers available for all courses. Because of this there are not enough teachers available to assess the practical tests by hand, so a system needs to be made that can assess the tests automatically. Assessment will need to be made of large groups of students, currently around 300-400 students. It is likely that this number will grow in the future.

1.2 The project

In 2007, during an internship at the UCSC in Shri Lanka, a system prototype was created that is able to assess certain Office skills of students. This prototype has been developed by Bert Wolters and Rick van Nierop. The prototype however has never been used, because it needed to be extended and improved. This year, CICAT, the central liaison office at TU Delft, has continued the Office skill assessment project (OSAP). This has been done as a part of the eBIT project. The goal of the eBIT project is to develop a 3-year Bachelor of Information Technology program. The University of Colombo School of Computing (UCSC), the Stockholm University (SU) and the Delft University of Technology (DUT) are currently working on eBIT.

The UCSC currently uses Moodle, an open-source course management system (CMS), to help educators create e-learning courses. The Office skill tests have been integrated into Moodle so that no separate system is needed to allow students to take Office skill tests. Also, several customizations have been made to make Moodle more suitable for the UCSC. The specific part of the project that this document is about concerns Moodle integration and customization.

In parallel to the Moodle-part of the project, the core of the Office skill system was extended and improved. For example the Office tests that can be taken are extended to include Excel and PowerPoint tests. These extensions are not discussed in this document.

2. Existing prototype

The old system is a prototype designed to automatically assess the Office skills of a student as described in the International Computer Driver License (ICDL) standard. This prototype has been developed by Bert Wolters and Rick van Nierop in 2007. The system however has only limited capabilities.
One of the limitations of the old system is that is can only assess the Word processing skills of the student. This has been done because these skills are the most useful for students. Also, the web interface is stand-alone and does not have login capabilities.
There are 2 types of users: teachers and students. These users have different abilities, listed below.

The teacher has the ability to:

- create a test
- edit a test
- delete a test
- get the answer written by the student
- get the student's results

The student has the ability to:

- take a test and view the result afterwards

The system has the following functionalities in order to assess the tests:

- obtain the teacher's solution
- obtain the student's answer
- comparing the former 2 documents and assign a grade to the result

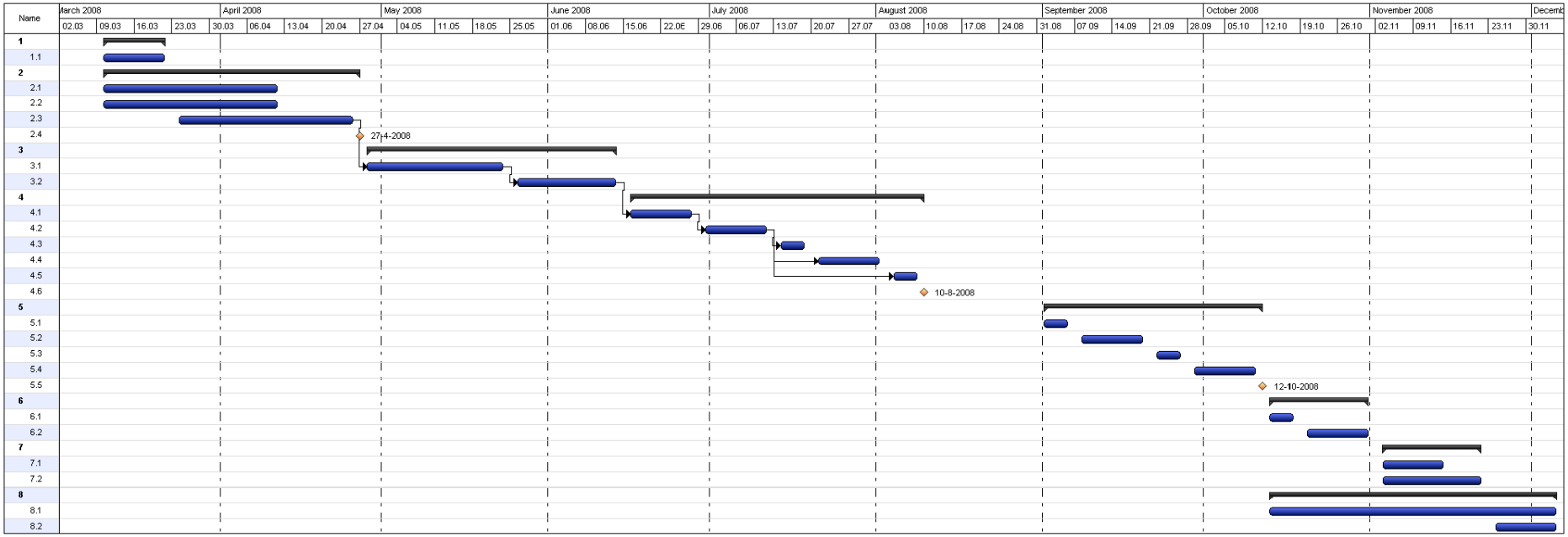
3. Project organization

3.1 Project planning

Activities
This project consists out of 8 different phases. Below each of these phases is mentioned and its tasks are described. The tasks mentioned are hard to understand at this point, but things will be made clear in the following chapters.

1. Planning
   1.1. Writing the workplan.
2. Analysis
   2.1. Getting the know the Moodle system.
   2.2. Getting the know the existing OSAP prototype.
   2.3. Writing the RAD (Appendix A).
   2.4. Release screen mock ups to the client
3. Design
   3.1. Writing the SDD (Appendix B).
   3.2. Writing the ODD (Appendix C).
4. Integration of OSAP into Moodle
   4.1. Combining the Moodle and OSAP database, make necessary code-adjustments.
   4.2. Make basic Moodle question type plugin, which will be expanded during the project.
   4.3. Integrate the old GUI into the question type plugin.
   4.4. Build the forms and required functions of the question type plugin.
   4.5. Moodle gradebook integration.
   4.6. Release prototype of question type plugin
5. Moodle customization
   5.1. Adapt the question engine to allow the random selection of a sub category.
   5.2. Add functionality to make the selection of following questions conditional.
   5.3. Ability to recover the cookie information stored at the client.
   5.4. The supervisor should be able to grant additional session time for tests.
   5.5. Release prototype of Moodle customizations
6. Ease of installation
   6.1. Make sure the Moodle OSAP plugin is easy to install.
   6.2. Write a installation manual for the OSAP plugin (Appendix D).
7. Testing
   7.1. Test the system.
   7.2. Write the testdocument.
8. Endreport and presentation
   8.1. Write the endreport.
   8.2. Give a presentation.

Milestones
27/04/2008: Release screen mock ups to the client
10/08/2008: Release prototype of question type plugin
12/10/2008: Release prototype of Moodle customizations



Note: work on task 8.1 (writing the endreport) will be done only on friday for the specified duration of 8 weeks

Phase
Task
Milestone
Link

3.2 Technologies

The following technologies will be used during this project:

- PHP (object-oriented)
- MySQL
- HTML

4. Introduction to Moodle

The UCSC currently uses Moodle. Moodle is designed to help educators create online courses with opportunities for rich interaction. Its open source license and modular design means that people can develop additional functionality. This chapter is meant as a short introduction to the Moodle features relevant to this project. Section 4.1 will briefly explain uses of the quiz module. After this, section 4.2 will describe questions types and other related features.

4.1 Moodle quiz module

The Quiz activity module allows the teacher to design and set quizzes consisting of a large variety of Question types, among them multiple choice, true-false, and short answer questions. These questions are kept in the course Question bank and can be re-used within courses and between courses. Quizzes can allow multiple attempts. Each attempt is automatically marked, and the teacher can choose whether to give feedback and/or show the correct answers.
Feedback on performance is a critical part of a learning environment and assessment is one of the most important activities in education. As educators, we can't tell what's going on inside the heads of students, so we need a way for them to demonstrate what they understand and what they don't. A well-designed test, even a multiple-choice test, can give you critical information about student performance. If the feedback is rapid enough, it can also be a critical tool for students to gauge their own performance and help them become more successful.
Moodle's quiz module has a large number of options and tools, making it extremely flexible. You can create quizzes with different question types, randomly generated quizzes from pools of questions, allow students to have repeated attempts at a question or retake quizzes multiple times, and have the computer score it all.
(http://docs.moodle.org/en/Quiz_module)

4.2 Moodle questiontype

Question bank
This feature allows a teacher to create, preview, and edit questions in a course question bank, a database of questions. These questions can then be used in Quizzes. The teacher enters the question bank by creating or editing a quiz activity or through the Administration block.

The initial Question Bank page has tabs that allow you (as teacher) to edit questions, categories, import questions and export questions.

Select a category
Questions are organized into categories. Initially each course has only one category called "Default". It is good practice to create more categories to organize your questions. You can create a hierarchy of categories because you can create subcategories inside parent categories. To add or edit categories click on the "Categories" tab.
The question editing screen shows the questions from the currently selected category. You choose this category from the Category: drop-down menu. Using the tick box below that menu you determine whether to also show the questions from all subcategories.

Add a new question
1. Click the Questions tab to access the Question Bank page, if not there already.
2. From the Category drop-down menu, select a category you want to add a question to.
3. The page will change to show the questions already in that category
4. Select the question type you want to create from the Create new question drop-down menu.
5. Fill in the form for the question type you are creating. Each question type has its own form and has its own options.
6. Click Save Changes at the bottom of the form.

Preview, Edit, Delete, and Move
The first column in the list of questions contains a number of icons and a selection box.
Clicking on the Preview icon will open a preview window in which you can test the question. The Edit icon allows you to edit the question via the same form that you used to create it. The Delete icon deletes the question, provided it is not already in use in some activity. The selection box allows you to select a subset of questions that you can then move to another category using the controls below the list of questions.
(http://docs.moodle.org/en/Questions)

Question engine
Moodle has a powerful question engine with a modular structure to allow question type plug-ins. The question engine is responsible for rendering the questions and for processing student responses. It is used by the quiz module and it is planned that in future it will be used by the Lesson and other modules.
(http://docs.moodle.org/en/Question_engine)

The question engine of the Moodle system allows the designer of the test to define categories and sub categories (groups of questions) and to generate randomly selected questions from the categories or (sub categories). For instance the test can consist of:

- 3 randomly selected questions from category A
- 2 randomly selected questions from category C – sub category 2
- 1 randomly selected questions from category B

Gradebook

Many activities in Moodle, such as assignments, forums and quizzes may be given grades. Grades may have numerical values, or words/phrases from a scale.
Grades can also be used as outcomes and as arbitrary text attributed to each participant in a course.

Grades pushed by modules

When activity modules produce grades, they use the gradebook public API to push (or send) their grades to the gradebook. These grades are then stored in database tables that are independent of the modules. The grades are still kept in the module database tables, and the gradebook will never access or modify these original grades.
The gradebook, however, provides administrators and teachers with tools for changing the ways in which grades are calculated, aggregated and displayed, as well as means to change the grades manually (a manual edit of a grade automatically locks the grade in the gradebook, so that the module which originally created the grade can no longer update that grade in the gradebook until the grade is unlocked).

Grades organisation

Teachers may organise grades into grade categories, import and/or export grades, and make grade calculations.
Symbols to represent ranges of grades may be set as grade letters.
Administrators may control the appearance of the gradebook site-wide by adjusting settings available via the grades link in the site administration block:

- General grade settings
- Grade category settings
- Grade item settings
- Gradebook report settings

(http://docs.moodle.org/en/Gradebook)

5. Analysis

5.1 Problem description

There are two main questions that need to be answered in this report:

1. How can users that are logged in into Moodle be allowed to make tests with Office skill questions?
2. How can Moodle be made more suitable for the UCSC?

In order to answer the first question we need to know a couple of things about Moodle. Moodle has a number of special activities, which educators can add to courses. When a student is enrolled in such a course, he or she can take part in this activity. An example of an activity module is the quiz module, which enables students to take tests consisting out of various types of questions. The quiz module can be extended to include various types of questions by the use of question type plugins.
Integration into Moodle can be done in a number of ways. The first method of integration is to create a new activity module for Moodle. This would mean that a large part of the current user interface needs to be rebuild from scratch. Also, mechanisms need to be created to allow teachers to create/edit tests and students to take tests.
The second method of integration is to make a new question type plugin for the quiz module. This way the quiz module can be used to build tests consisting out of Office skill questions. This means the overhead of creating a separate activity module would be eliminated.
The answer to the second question consists of a number of customizations to the Moodle code.

5.2 Core functionalities

Like the current system, the new system recognizes 2 types of users: teachers and students. The abilities of these users are listed below, followed by the user authentication and assessment part of the system.

T1 The teacher is able to create an Office skill question. He/she can:
- T1.1 Select the category to which the question belongs to.
- T1.2 Set an identifier for the question.
- T1.3 Add the actual question, which consists out of:
  - T1.3.1 Instructions for the question.
  - T1.3.2 An Office Open XML (OpenXML) file representing the correct answer.
  - T1.3.3 A marking scheme.
- T1.4 Set the maximum answer-size that a student can upload.

T2 The teacher is able to edit an Office skill question (see T1.1-T1.4).
T3 The teacher is able to delete an Office skill question.
T4 The teacher is able to add an Office skill question to a quiz (see section 3.1).
T5 The teacher is able to sets a time limit for a quiz.
T6 The teacher is able to download the most recent attempt made by the student.
T7 The teacher is able to view the results of students in the Moodle gradebook.

S1 The student is able to make a test with Office skill questions. He/she can:
- S1.1 Sign in to a test.
- S1.2 View the remaining time given to make the test.
- S1.3 Get a warning if the time left to make the test is less than 5 minutes.
- S1.4 Get a message if the time has expired and submit whatever questions have been answered.
- S1.5 Upload a question answer (in Office Open XML format).
- S1.6 Submit the question answer and proceed to the next question.
- S1.7 View the result of the submitted test.

S2 The student is able to view his/her grades in the Moodle gradebook.
S3 The student is able to take practice tests and final tests.

The system has the following functionalities in order to deal with different users:
U1 The system is able to authenticate users.
U2 The system is able to determine whether a user can perform a certain task.
- U2.1 A student can perform only the actions described in S1-S3.
- U2.2 A teacher can perform only the actions described in T1-T8 plus all actions that a student can perform (S1-S3).
- U2.3 If the an attempt is made to perform a unauthorized action, display an error message.

The system has the following functionalities in order to assess the tests:
A1 Obtain the teacher's solution, which consists out of:
- A1.1 The data from T1.3.1-T1.3.3.

A2 Obtain the student's answer, which consists out of:
- A2.1 An Office Open XML (OpenXML) file representing the students answer.

A3 Make sure that a student does not spend more time than is specified.
A4 Assign a grade, using:
- A4.1 The documents from A1 and A2
- A4.2 Expert data about what should be considered as right and wrong.

The priority of these functionalities can be considered ‘must have’, according to the MoSCoW method.

5.3 Customizations

The question engine of the Moodle system allows the designer of the test to define categories and sub categories (groups of questions) and to generate randomly selected questions from the categories or (sub categories). For instance the test can consist of:

- 3 randomly selected questions from category A
- 2 randomly selected questions from category C, sub category 2
- 1 randomly selected questions from category B

This functionality is to be expanded to include the following:

Q1 Additional functionality to the question engine is needed is to randomly select a sub category.

The old situation is as follows:
Moodle supports adding random questions to a quiz. These random questions, when loaded, merely link a another question. The question to which is linked is randomly chosen from a pool of questions. The questions within this pool are either from a category and its subcategories or only from a category itself.
What questions to put in the pool is determined in the questiontext attribute from the question table in the database. A 1 means choose question from the category and its subcategories, 0 means only use questions in the category itself.

Excluded from this pool are the following:

- All questions that are explicitly assigned to the quiz
- All questions that are already chosen by another random question
- Random questions
- Other explicitly excluded question types
- Wrapped questions

The random questiontype class has a property 'catrandoms' which is an array indexed by category id and by $question->questiontext. Each entry is a randomized array of questions in that category which can be used.
The class also has an additional property $quiz->questionsinuse that holds a comma separated list of all questions used in the current quiz.
(http://docs.moodle.org/en/Random_question_developer_docs)

Q2 The question engine should be adapted so that questions from the sub category can be selected randomly (existing functionality), but with the option to make the selection of following questions conditional (this does not exist but is required). Example of conditions: if question K is selected, than it should be followed by question L and M. Another example: if question C is selected than question A and L should be excluded as following questions.

Besides these customizations to the question engine, the following functionalities are also required:

Q3 The supervisor of the test facilities should be able to grant additional session time to students in case of an event out of control of the student and resulting in a loss of time.

The old situation is as follows:
A quiz can have a time limit. This is stored in minutes in $quiz->timelimit. So before using this in time calculations it always has to be multiplied by 60 to turn it into seconds like all other timestamps in moodle and php. If $quiz->timelimit is zero it means there is no timelimit.
The time a response was submitted by the student is recorded by attempt.php right at the top of the page.
(http://docs.moodle.org/en/Quiz_developer_docs#Time_limit)

Q4 Another functionality needed is the ability to recover the cookie information stored at the client stations in case of an unexpected shut down of the client PC. This way the user at the client PC will not need to re-login when the PC is restarted.

The old situation is as follows:
Moodle, by default, uses cookies in order to track user sessions. The cookie that is responsible for this is called MoodleSession. The data belonging to a session is stored in a file inside the Moodle dataroot. Without the MoodleSession cookie, there is no way of knowing what session belongs to which user. Because of this, there is no way to restore such a cookie from within the Moodle system.
So, now we know that it is not possible to restore a cookie, the next best thing would be to disable cookies altogether. Moodle does support cookieless sessions if this is explicitly mentioned in config.php. When this option is active the session ID (used to track the user session) is made a part of the URL. Most browsers remember the URL that was visited, so that when the browser is restarted, the same page is loaded again. However, cookieless sessions are only used when the browser does not support cookies.

The priority of these functionalities can be considered ‘should have’, according to the MoSCoW method.

6. Design

In this chapter the methods of tackling the problems from chapter 5 are discussed. Section 6.1 will present a way for the Office skill system to be incorporated into Moodle. And after this, section 6.2 will discuss the design of the customizations to Moodle.

6.1 Office skill plugin

A new question type plugin was designed in order to integrate the system into Moodle. This section briefly discusses some the design characteristics.
The officeskill_qtype class is one of the most important classes. For example, it pushes grades to the gradebook and handles user responses. It does this by extending the functions of the default_questiontype class, as shown in the image below.

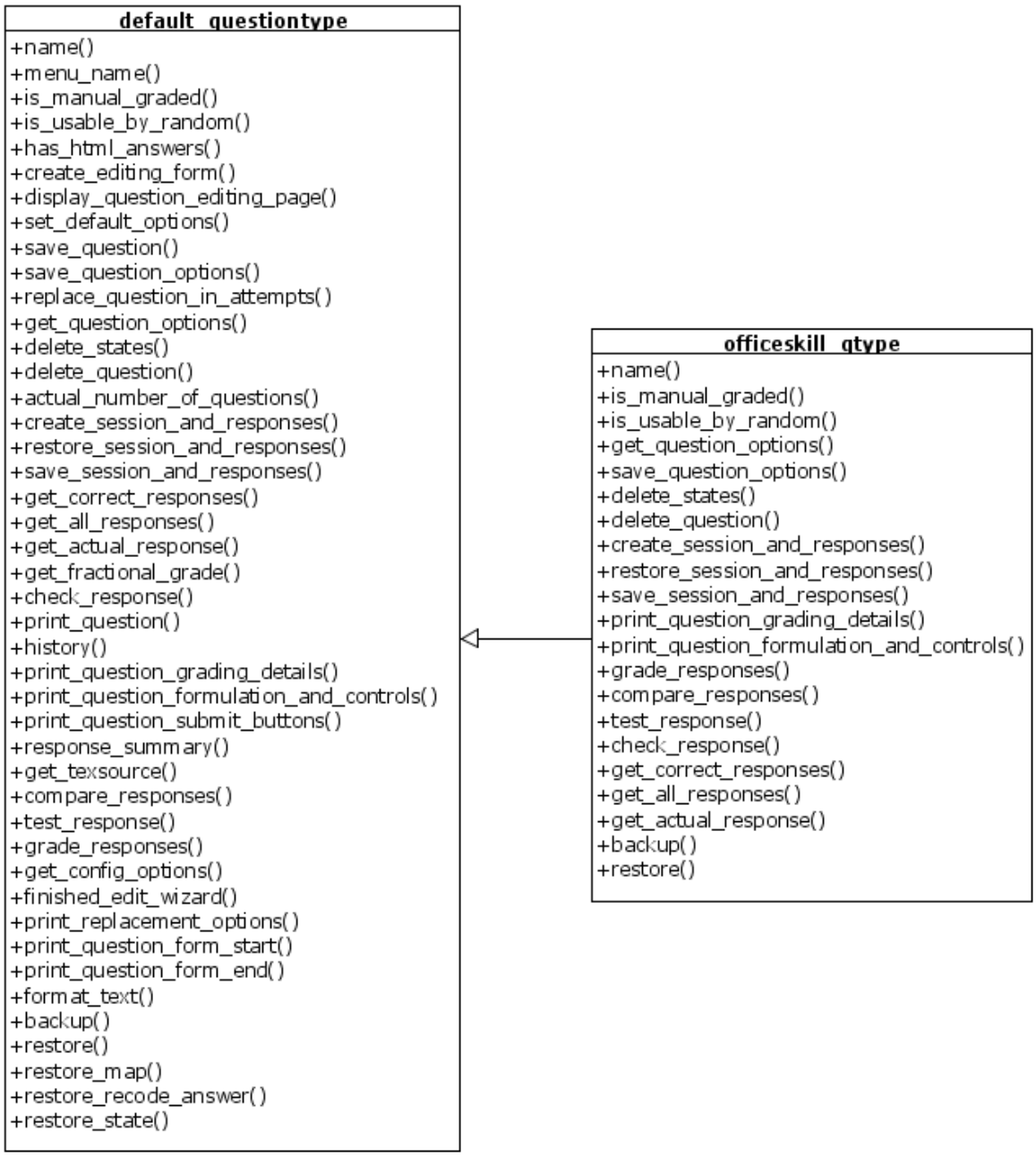


Figure 1: Office skill question type class

Processing of responses:
The processing of the responses takes place in the officeskill_qtype class. First, the question itself is printed by the print_question_formulation_and_controls() method. After the question is printed, the system only processes it again when the question is submitted. When this happens, the grade_responses() method is called in order to upload the student's file to a temporary file located in the Moodle data directory. When the upload is completed, the function passes the file on to the DocumentHandler class, that parses the file. After parsing, the temporary file is either deleted (in case of an error) or made permanent (if the parsing was successful). After that a grade is received from the DocumentHandler and this grade is passed on to Moodle. The compare_responses() function is used to determine if the current response is identical to the previous one. This way grading does not have to start again, if the same response is given repeatedly.

Downloading of responses:
The downloading of the most recent solution of a student is handled by question/type/officeskill/downloadsolution.php. The script checks if the user is logged in and has grading capabilities for the quiz.

Figure 2 shows a part of the database structure of the Moodle question engine and the Office skill database. The Office skill system needs to use the Moodle database design if it is to be a question type plugin. The question table contains attributes common to questions of all question, and the question_officeskill table contains a few attributes specific to the Office skill question type.

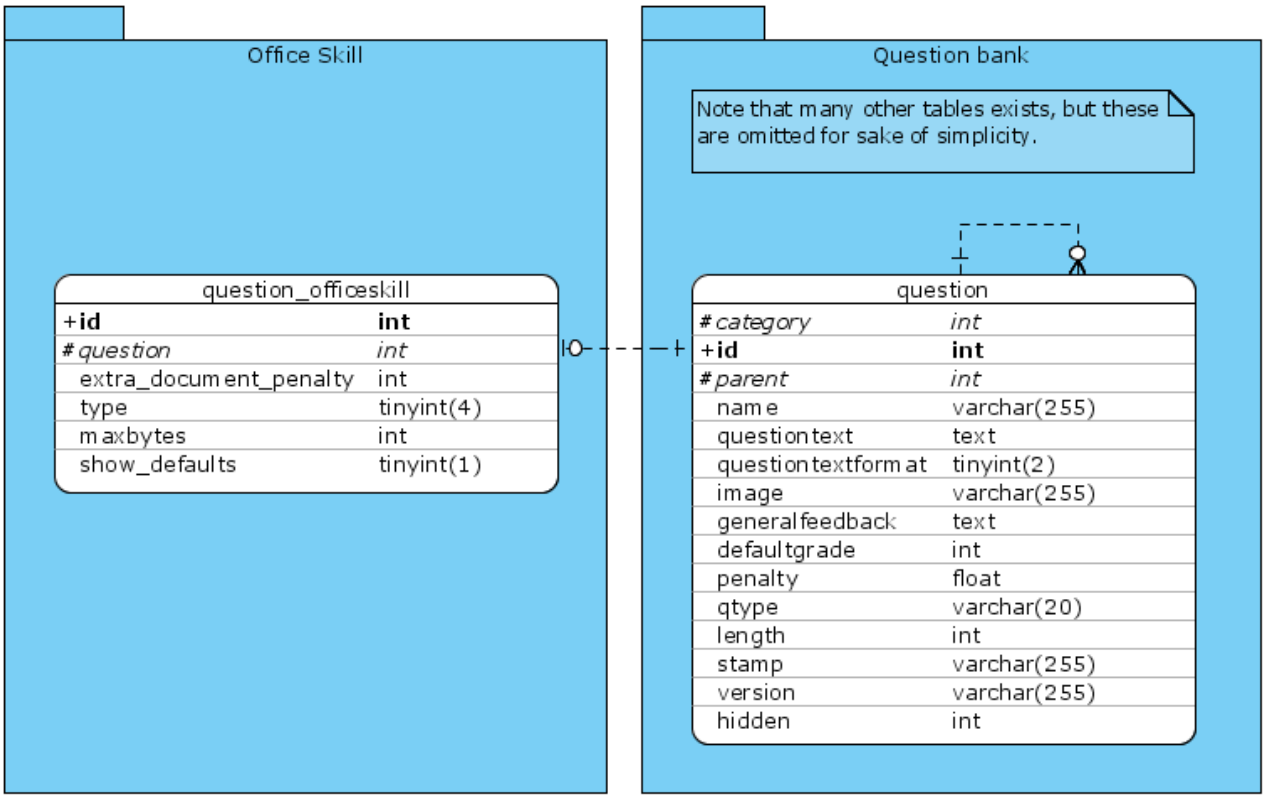


Figure 2: Office skill database

The interface for creating or editing an Office skill is shown in figure 3. This interface is to be build using the question_edit_officeskill_form class, which uses the Moodle forms library. Most of the entries in this form should look familiar to those that have used the quiz module before. Only the last three entries are specific to the Office skill questions.

After creating or editing an Office skill question (figure 3) the user is redirected to the testeditor. This is done by setting the variable $SESSION->returnurl at the end of the save_question_options function in the officeskill_qtype class.

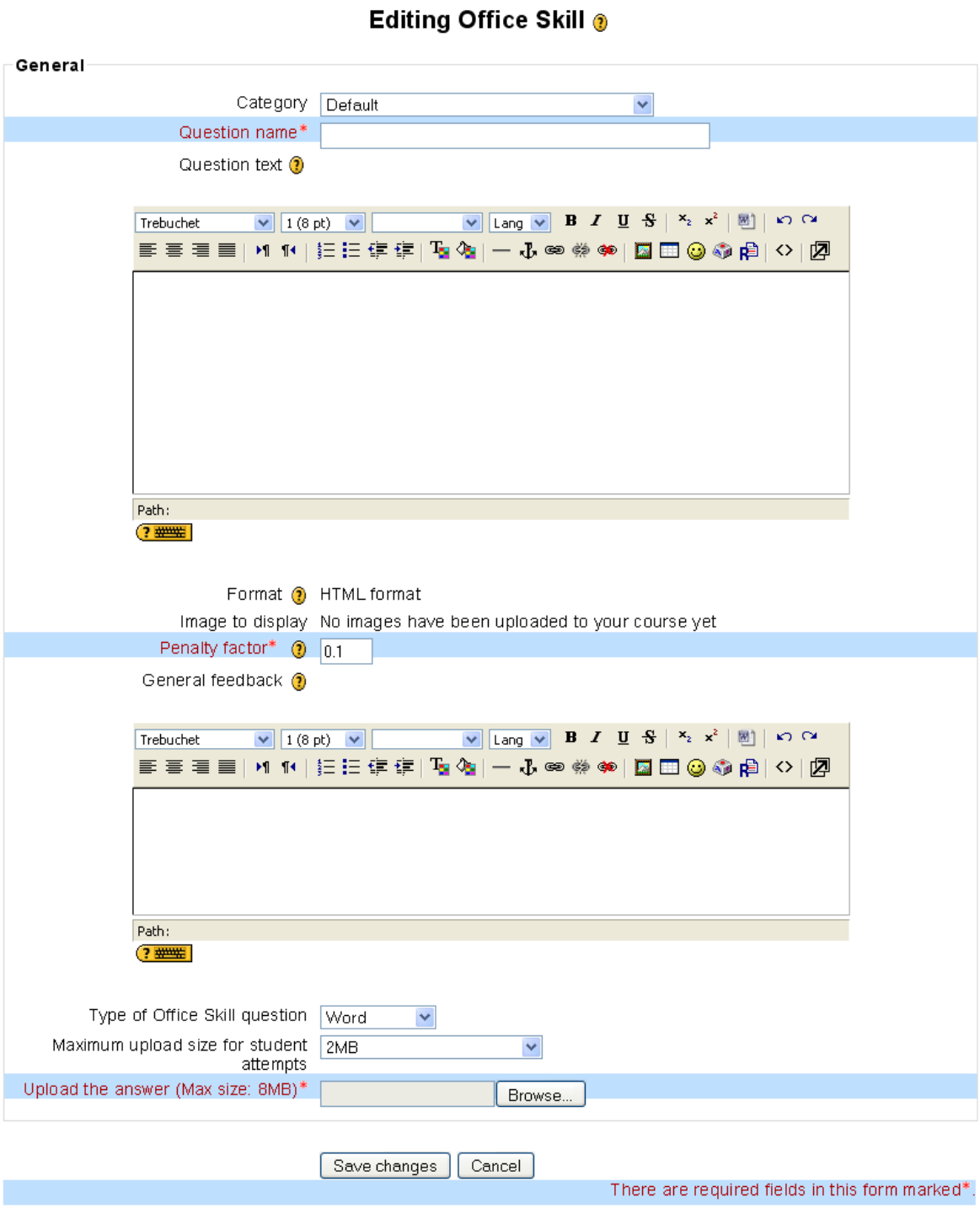


Figure 3: The teacher interface

The student interface is shown in figure 4. When a solution is uploaded a message indicating this will be displayed right below the upload field.

Figure 4: The student interface

6.2 Moodle customizations

Random selection of subcategories (Q1)

In order to add the required functionality, the use of the questiontext attribute was expended. While the functionality of the first character remains the same, the second character from the questiontext attribute is now used to determine whether to select the question from a random subcategory. The character is set to 1 for the random selection of a subcategory, a 0 or nothing means don't select a random subcategory.
When a number of random questions are added to a quiz with the random subcategory checkbox set, this group of questions will all be selected from the same subcategory. In order to achieve this grouping an attribute called groupid has been added to the quiz_question_instances table. Questions that are added at the same time will all have the same groupid.
The pool of questions from which to select the random question is the stored in catrandoms[0][$questiontext.$groupid].

Question conditions (Q2)

An extra table called question_conditions has been added to the database (figure 5). Each entry in this table represents a condition. A condition exists between two questions, represented by the foreign keys question1 and question2. The type attribute is either a 0 for a include condition or a 1 for a exclude condition. The management of these conditions can take place using the "Question conditions" page of the quiz edit tab. The files question/condition.php and question/conditionlib.php are be used for this.

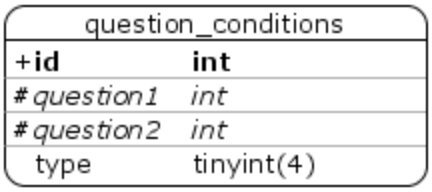


Figure 5: The question conditions database table

The file question/type/random/questiontype.php should be altered to support question conditions. When a potential random question is chosen, there are a few rules that need to be uphold:

- The question cannot be excluded by any of the questions that have already been chosen.
- The class Conditionsmatrix can determine the length of a chain of conditions. For example if A includes B and B includes C, then the length of the chain is 2. If A includes B and B includes C and B includes D, then the length of the chain is 3. The chain length belonging to a question must not be greater than the number of questions that still need to be selected.
  When this condition is met, the next questions that will be chosen will be the questions from the chain.

By default, Moodle processes the questions in the order that they are found in the database. However in order for this customization to work we need Moodle to process the questions in the same order as they are displayed on the screen, in order to achieve this, some alterations need to be made to mod/quiz/attempt.php.

Assigning extra time to quizzes (Q3)

An attribute called extra_time needs to be added to the quiz_attempts table in order to store the amount of seconds of extra time is given to a certain attempt at a quiz. This extra time is then added to the timelimit for the quiz. This can be done in mod/quiz/attempt.php, near the end of the file.
The extra time can be set from the overview page of the quiz results tab.

Recovering cookies (Q4)

With some customizations to the code cookieless session can be forced, so that cookies are not used even if the browser does support them.
However with this design there are also some security considerations. Mainly, session IDs in the URL are a bad idea because people like to bookmark pages. It is not desired behavior that users keep using the same session ID, or maybe even use the session ID of another user.
Overall you could say that this customization would cause more problem than it solves, and the idea has therefore been discarded.
Restoring a cookie should be done by the browser. Firefox, for example has a feature called session restore (https://wiki.mozilla.org/Session_Restore), which restores cookies when the PC has unexpectedly shut down.

7. System testing

In this chapter the testing will be discussed. Testing is separated as follows:

- Unit testing: The goal of unit tests is to ensure that the functions and methods of classes individually, are working properly. It is one of the first steps in a quality control process for developing code. The unit tests were made using the SimpleTest framework. SimpleTest is incorporated into Moodle. Unit testing in Moodle is described in section 7.1.
- Functional testing: Testing of the functionalities from chapter 5. This is discussed in 7.2.
- Pilot testing: This is to test the actual system in the field. This is discussed in 7.3

7.1 Unit testing

For the unit testing a separate test case class was made. This class was written as an extensions of a base test case class provided by the Moodle system. The test case class consists of a number of methods that contain the actual test code. Each of the methods invokes various assertions that are expected to be true or false. For example the assertion *$this->assertEqual($this->qtype->name(), 'officeskill');* passes if $this->qtype->name() returns the string 'officeskill'. If all the methods of the test class are run without problems, the test is considered successful.

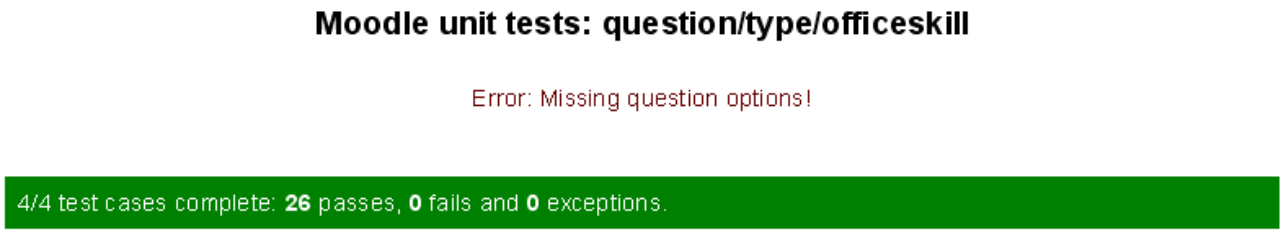


Figure 6: A successful test run

After a successful test run the user is presented with a report as shown in figure 6. The figure shows an error, but the user is told that all test cases have passed. This is because the testcode tries to retrieve a nonexisting question from the Moodle database using a assertFalse assertion. Please ignore this error.

Tests on Moodle can be run as follows:

1. Login as an administrator.
2. On the left of the screen there is a menu called 'Site Administration'. In this menu, click on the 'Reports' link.
3. Click on the 'Unit tests' link.
4. Click on the 'Run tests' button to run the unit tests (figure 7).

If you run the tests as mentioned above, all parts of the codebase are run. However, if you would just like to run the test code belonging to the Office skill questiontype you can type the name of the Office skill directory (question/type/officeskill) in the textfield (figure 7).

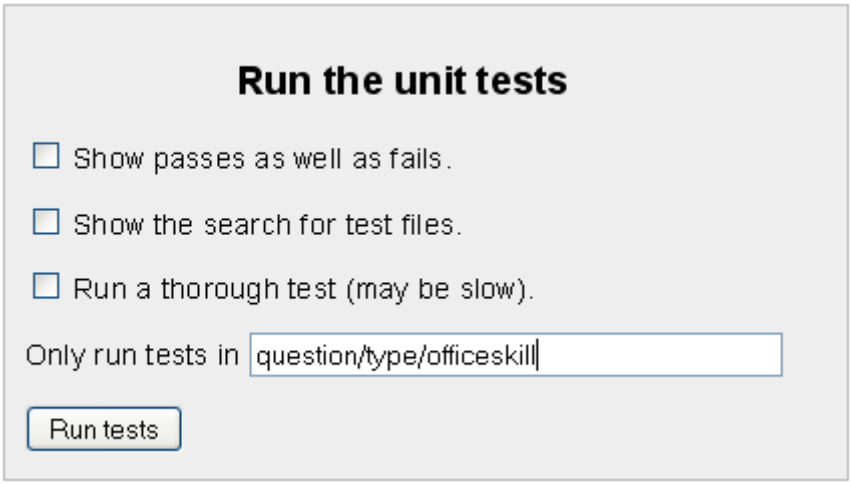


Figure 7: Run the officeskill unit tests

The most interesting problem that was found were some 'file not found' errors when the system was used on Linux. The problem was due to the fact that Windows and Linux use different directory separators. The fault (located in the core of the Office skill system) was fixed.

7.2 Functional Testing

Comparing the implemented functionality with the described functionality from the RAD (Appendix A), you can see they are in large part the same. However, there are some differences:

- In the implementation there is no distinction between practice tests and final tests (requirement S3). This distinction, however, is unnecessary because a teacher can just create two separate quizzes, namely one for practicing and one for final tests.
- A teacher is supposed to be able to perform all actions that a student can (requirement U2.2). However, the teacher is not able to view his/her results in the gradebook, since the teacher cannot get grades. The teacher is able to preview the quizzes.
- In the RAD the form for editing an Office skill question contains the testeditor at the bottom of the page. In the implementation, the testeditor is located on a separate page from Office skill question edit page.
- In the RAD the form for creating/editing an Office skill question contains a field to set the extra document element penalty. Since this field already is present in the testeditor, the field has been removed from the question creating/editing form.
- The link to the most recent answer of a student is not located on the results page of a quiz (as it is depicted in the RAD), but is now located at the grade details page.
- Cookies are not recovered by Moodle (requirement Q4). In chapter 5 this is discussed in further detail.

Other then the remarks described above everything functioned as described.

During the functional testing some bugs were also found. Some of these bugs are:

- When the user pressed the cancel button in the testeditor, a message was displayed indicating that changes were made to the database.
- The restoration of a backup of the Office skill questions did not work properly.

Initially, testing was done on Moodle 1.8.1. After that the Office skill question type was also tested with Moodle 1.9.3. It was suspected that everything would work, however, it turned out that there were some errors:

- The new version of Moodle uses a different way to store the URL to which to return after saving a question. Since the testeditor needs to be opened after saving a question, this no longer worked.
- Moodle 1.9.3 added an extra language string for adding a question.
- The security measures did not work because Moodle 1.9.3 did not store the courseid in the question_category table anymore.
- The upload feedback that a student was presented with after an answer was submitted, did not function anymore.

All the bugs mentioned above have been fixed.

7.3 Pilot Testing

This will be done for the entire Office skill system by a number of users at the UCSC. The system was not completed at the time of writing, and the testing will be published in a separate report.

8. Conclusion and evaluation

8.1 Conclusion

The original assignment consisted out of two parts. The first part involves the integration of the Office skill system into Moodle. The main goal here is to allow users that are logged in Moodle to make tests with Office skill questions.
The second part consists out of a number of customizations of the Moodle code to make Moodle more suitable for the UCSC.
Looking at the result of the project, it can be considered a success. The Office skill system is now fully integrated into Moodle. Students logged in can now take part in quizzes consisting out of Office skill questions.
The customizations for Moodle were also successfully completed, with the exception of ability to recover the cookies used by Moodle. Since this task is not really up to Moodle, this is not a big loss. It is recommended that the client looks for a browser that supports the ability to recover cookies.

8.2 Evaluation

This project was a success. A solution to the problem was found and implemented. Looking back at the project, however, there were some things that could have been done better:

- At the beginning of the project, the implementation of a solution started too early. The first solution was to implement an entire new module for Module. However, it turned out that implementing a new questiontype plugin sufficed. With this plugin the quiz module could be used to let students take tests. Unfortunately some time was wasted because of this.
- Development of the question type was done using Moodle 1.8.1. Testing the question type plugin with Moodle 1.9 started too late. It turned out that there were a number of problems with Moodle 1.9. Fortunately, these problems could be fixed in time.
- The communications with other members of the project were not optimal, especially towards the end of the project. This was because towards the end of the project my tasks were less correlated with the tasks of the other project members.
- A couple of months inwards, some new requirements turned up. These new requirements included some customizations to the Moodle code. This was not planned for, initially. Fortunately, there was enough time to do these customizations as well.
- Because the core of the system was being extended in parallel to my part of the project, the integration needed to be restarted a number of times. This was necessary because the changes made to the core influenced the way the question type plugin worked.

Luckily, a number of things did go well:

- To start with everything works well. The client was happy with the idea of a question type that extended the use of the quiz module.
- The assignment itself was clear, leaving no room for misinterpretation.
- The planning was met almost to the day.

9. References

- A comprehensive reference of all known Moodle modules, hacks and plugins
  http://www.moodle.org/modules
- Documentation for Moodle
  http://docs.moodle.org
- SimpleTest
  http://www.simpletest.org/
- The source code and technical documentation of the old system is available at (password-protected):
  https://cpsvn.twi.tudelft.nl/svn/osap

Appendix A: Requirements Analysis Document

Appendix B: System Design Document

Appendix C: Object Design Document

Appendix D: Installation procedure