The developments in Computer Based Education at the Queensland University of Technology over the past three years have been driven by advances in Instructional Technology and the increased capability of the technology to respond to the needs of students.

The aim of this workshop was to examine some of the issues facing us in trying to deliver quality education and to demonstrate some of the ideas being developed in the use of teaching technology at QUT to deal with such issues.

The notion of the computer as being primarily useful for assessment has reduced somewhat at QUT however its use in formative assessment has increased. Decision-making programs are being used for formative assessment and feedback is being either built in to programs or being provided outside of the computer system. Examples of the type of software we have been moving toward as well as the context of its use was examined.

**Examination of some computer software used for assessment purposes in QUT**

**Anatomy & Physiology**

The Anatomy & Physiology software is essentially a question and answer package based on a variety of topics in this subject area. The question types are:

- multiple choice with option specific feedback
- true/false
- text based short answer
- graphic based short answer

Question sets of up to 30 questions are generated from a larger set which is stored in a database. When the student is presented with the question set they are tracked and results stored.

The multiple choice questions were designed so that all alternatives 'trapped' common misconceptions and the feedback provided for each alternative clarified these misconceptions.

Guidelines used in designing these types of questions included:

- only important facts and knowledge were tested
- lifting statements verbatim from the text was avoided
- only one correct answer was available
- negatives were avoided
- no clues were given
- all questions were reviewed
- focus was kept on the purpose of the test
- graphic design was kept consistent for all graphics based questions

These results may be used for formative assessment or for summative assessment. It has been found through focus groups and surveys that unless some form of summative assessment is attached to the use of this software — students will generally not use it.
To promote its use without actually using results for summative purposes, lecturers based questions in the final examinations on the types of questions in the database and informed students of this. Students then saw this as a useful resource.

Microcraft Author was used to develop, deliver and manage the tests and Borland DBASE 3 was used to store all test items.

**ChemPak**

ChemPak was designed for use by foundation year students to practise their problem solving skills in Chemistry. It contains a finite set of question templates from which questions are generated. Multiple step questions are linked together to form a ‘problem’ and the student has the capacity to reset a problem with a new set of parameters in order to practice that problem type. Solutions can be viewed by the student at any step of the problem solving process.

The software can be set to practice mode or assessment mode — which allows the software to be used as a learning tool as well as an assessment tool — either formative or summative. In assessment mode the solutions are not visible to the student.

All questions are generated during the running of the software by drawing parameters from a database system. This means that at different runs of the software, different versions of a problem type will be presented to the student. Students tend to develop a “try to beat the last score” attitude to the problems.

Student feedback has been extremely positive and confidence in problem solving strategies for Chemistry seems to be increasing. Evaluation studies are being conducted.

Asymetrix Toolbook IV was used to develop the software and a Microsoft Access database system holds the data for question generation.

**Committed**

The Committed software is a simulation which was designed to run throughout a semester to provide a management ‘experience’ for students studying Construction Management. Its demonstration in this forum was to illustrate an alternative assessment strategy that may be applicable to some units in Science courses.

During the course of the semester, students work in teams to address the issues which occur during a simulated construction project. They make decisions, monitor budgets, schedule tasks, select resources and so on. While working through the events which occur during the simulated construction project, students develop a final report. This report is the item of assessment that is submitted for summative purposes however during the process, students are given guidance and feedback in a formative manner.

The report is marked according to a scheme that does not have heavy weighting on the financial outcome of the simulation, rather the weighting is distributed across areas such as reasons for making decisions, expectations of simulation outcomes, students interpretation of client attitudes and the organisation of the group ie degree of cohesion, critical thinking etc. Students are made aware early in the process of this marking scheme and also of the nature of the report — they are told that it is to be a “candid review of the group’s experiences in working through the simulation”.

After students submit final reports, a final tutorial session is conducted during which the simulation is ‘acted out’ to a conclusion by lecturers. A further short test is conducted after the final report is submitted to allow differentiation between individual students in each group.

Similar teaching systems could be developed for use in Science related units. This would involve structuring the unit around a ‘project’ or ‘research simulation’. The collaborative team based approach would form a core part of the process and be assessed along with the content and processes used.

The program was developed using Asymetrix Toolbook 3 and Borland DBASE 3.
In conclusion, it is important to note that many factors contribute to the selection of any assessment system. If the system is to be of a high quality, it requires:

- a variety of methods aligned to learning outcomes;
- clear and unambiguous questions;
- consistent marking criteria which provide direction to both students and staff;
- clear constructive feedback provision (formative assessment);
- items which promote deep learning strategies.

Given that no single assessment type will provide for all of the above, it is clear that the aim is to integrate elements into a coherent and manageable ‘system’ which satisfies all of these requirements.

Before adopting or developing a computer based assessment system, it is recommended that a thorough analysis of the current system be undertaken. A checklist of questions (Appendix A) is an invaluable tool for evaluating the worth of any computer based assessment strategy.

Adoption of computer based assessment should occur only when it is appropriate for the establishment of a suitable mix of assessment elements which when combined satisfy the requirements of a high quality ‘assessment system’. The power of computers as a tool to individualise instruction with interactive strategies may in fact override its power as a tool for assessment.

Appendix A

<table>
<thead>
<tr>
<th>Checklist for evaluating a computer based assessment strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Does the strategy add to the variety of assessment methods currently in use? eg. a new style, competency based, collaborative method</td>
</tr>
<tr>
<td>2. Does the strategy provide or provide for clear and unambiguous questions?</td>
</tr>
<tr>
<td>3. Can questions be altered after reviews?</td>
</tr>
<tr>
<td>4. Is there a practice mode? ie. is formative assessment possible?</td>
</tr>
<tr>
<td>5. Is there an efficient way of avoiding/preventing plagiarism?</td>
</tr>
<tr>
<td>6. Does the strategy provide flexibility in allocation of time allowances for answer entry?</td>
</tr>
<tr>
<td>7. Are the marking criteria obvious, useful and matched to objectives?</td>
</tr>
<tr>
<td>8. Is feedback suited to the student and learning outcome being assessed?</td>
</tr>
<tr>
<td>9. Is feedback positive, timely, explicit, focused, informative?</td>
</tr>
<tr>
<td>10. Is feedback tailored to the developmental stage of the students?</td>
</tr>
<tr>
<td>11. Does feedback contribute to learning and subsequent assessment?</td>
</tr>
<tr>
<td>12. Do the assessment tasks contribute to the totality of assessment tasks in a worthwhile way?</td>
</tr>
<tr>
<td>13. Is there a balance of theory and application?</td>
</tr>
<tr>
<td>14. Can all topics be assessed?</td>
</tr>
<tr>
<td>15. Could student questions be incorporated?</td>
</tr>
<tr>
<td>16. Can other assessment items be incorporated with this as a system?</td>
</tr>
<tr>
<td>17. Does it contribute to the total assessment system in a balanced way?</td>
</tr>
</tbody>
</table>

References
An Assessment Suite for Authoring and Reporting on WWW-Based Exams, Questionnaires and Quizzes

Craig A. Burton
Engineering Multimedia Unit, University of Melbourne
caburt@ecr.mu.oz.au

Robert W. Barnett
Department of Electrical & Electronic Engineering, University of Melbourne
r.barnett@ee.mu.oz.au

Abstract - This paper reports on the most recent use of a World Wide Web (WWW) based tool under development for creation of assessment pages for students. The software, the Networked Assessment Toolkit (NEST), has been employed to compliment existing on-line material for students such as laboratory class lists, illustrated lecture material and course information. These latter resources are readily produced from existing word-processor documents but assessment via the WWW was not approached before NEST was developed. Via a simple command language, NEST builds examinations that run via the WWW in a laboratory of computers. Responses are collated, marked and disseminated. NEST has been employed to run student feedback questionnaires and tutorial quizzes. For this trial, student evaluation of NEST and all on-line materials was favourable and the tedious and error-prone nature of exam marking was removed.

Introduction

This paper describes further experiences gained and issues arising from the use of the World Wide Web (WWW) as a means to communicate with, deliver course materials to, and assess students. The development of a suite of tools called the Networked Assessment Toolkit (NEST, commenced in 1995) automates the WWW collection of data from students. NEST has been deployed in numerous Engineering topics but here we focus on its most recent use amongst on-line materials in two particular subjects of a course at the University of Melbourne, in 1996.

The first subject, Biomedical Engineering, was presented to a class of 75 students. The second subject, Electronics, was offered to a much larger group of first year engineering students prior to their choice of specialisation. In this case, the class size was over five hundred.

Implementation

No student in either group had previously encountered WWW-based course content. It has been assumed, therefore, that their evaluation of the medium is generally unbiased. The lecture notes, tutorials and laboratory schedules were presented in the same “traditional” way as in previous years. In other words, students could attend “normal” lectures and tutorials armed with material obtained in advance from the library. Similarly, special announcements and lab schedule updates could be obtained either in lectures or from a notice board. In this way, students had free choice to ignore the computer medium if they wished and still complete the subject without hindrance. The great majority were quick to recognise that a computer medium offers a natural advantage. Information can be accessed instantaneously without the constraints imposed by timetables and library hours. Major examinations were still run on paper, but smaller exams were run exclusively via the NEST system. Students could elect to use a paper version of WWW assessment pages if they so chose.

The core of NEST functions as a compiler — it accepts a text file of commands and content and rewrites them as an executable program for interactive communication with the user’s browser program (such as Netscape). Since NEST was first trialed in 1995, it has removed the necessity for fluency in programming languages in order to produce interactive WWW documents. The language NEST uses is called “Dotty”, and is based around simple ‘dot-commands’. A simple example of which would be
What is spanikopita?
and is interpreted by NEST as requiring a text question “What is spanikopita”, an accompanying field for text (a %d type), and then assessment of whether the user entered “food, foodstuff, food*” etc, correctly, for 10 marks (zero otherwise).

Commands in Dotty have evolved since their first deployment. There now exists a larger suite of formatting commands for tables, text alignment, images, movies and data entry checking. A description of the language Dotty can be found at http://www.ecr.mu.oz.au/~caburt/nest.html

Following are descriptions of the implementation of assessment materials, their appearance to users and the process of authoring assessment content.

A. NEST Quizzes

For the purpose of on-line tutorials, NEST was used to run the 1995 final exam (Electrical Engineering 1995 November) as a WWW page. In this case, the existing exam was re-written as a Dotty file, with images from the exam converted from the original Word document. Below is a section of the exam demonstrating how a sketch response on the paper exam appears as a multiple choice item that allows a student to choose a graph. In this case, new images were needed.

Old and new: The engineering paper exam (left) was re-written in Dotty to produce a NEST quiz (right). To answer the paper exam question, students were asked to sketch a graph. In NEST, they must choose from a selection of graphs.

Students could gain access to this quiz by using a WWW browser such as Netscape from home or on-campus. They were requested to supply a username and password. Currently, all NEST users are also account holders in the central Engineering computing facility, so their NEST passwords are obtained from this central system.

The quiz appears as a WWW page containing a form with a submit button. The student can select or enter answers as appropriate and then submit the form. Different activities include multiple-choice, multiple-responses or production answers such as key entry of numbers or words.

The reply from the system is another page with a table created on the fly containing a basic marking scheme for the student. The table lists each individual response and its corresponding mark. Below is a section of such a table. There is no correct answer given but the student is offered a return button to go back to the quiz and try again if necessary.
Quiz results: A student is given this table when they submit a quiz with one or more answers entered. The results do not give answers, but assign a mark (in this case zero or one, but richer marking values can be written with Dotty).

Every submission that a student makes is recorded within NEST, along with the times for each data item submitted. We have noted that some students make very many submissions to quizzes in one sitting, while others achieve near-perfect scores with each submit, in progressive sections of the quiz, over several sittings.

B. NEST Exams

Two NEST examinations, comprising 15% of the total subject assessment, were compiled and implemented for the smaller class (75 students). The facilities available allowed each student in the class to sit the test simultaneously in the same building. No attempt was made to test the larger class in this way due to the administrative problems involved. However, an exam for the larger subject is proposed for this year which will utilise facilities throughout the university.

A NEST Exam appears much like a NEST quiz, except for extra security, a time-limiting facility and the absence of immediate feedback (such as marking). These exams are intended as “real-time” exams in as much as they are taken by groups of students in laboratories that are properly invigilated. Students do not gain access to these exams via links from other WWW documents, but must instead log into and hence set-up their exam terminal via a special username and password. A login script limits what programs they can access from the terminal. In this case, only “Calculator”, “Clock” and the browser could be accessed, but future exams might allow design or analysis tools such as MATLAB and SPICE. The browser, a modified version of NCSA Mosaic, automatically launches the exam page and denies “surfing” to any other WWW pages.

The exam appears to the student much as the quiz (above) does. In this case, however, the students are informed that they have a limited time in which to complete the exam. After this time, they can still submit, but if their total duration exceeds the allowed duration of the exam, they will be penalised. This passive approach was felt to be better than hard-limiting exam times in case students were late, or there were network delays. Exam open and close times are specified in Dotty. A student attempting to get in early is shown a screen informing them that they are early.

Students can submit what they have done, complete or not, at any time. Within the Dotty commands for the exam, it is possible to describe whether answers are compulsory, suggested or neither. An
unattempted compulsory item gives a bold warning to the student on the reply page, and the student is offered a button labelled with the unanswered question number. Clicking on this button takes them back to the exam, to the position of the corresponding unattempted item.

Reply from NEST: When specified in Dotty, NEST will inform the user that they have missed certain items and direct them back to the original document.

Submitting an exam with all compulsory items filled but some “required” items empty results in the student being informed of a complete submission, but offered return buttons for items that they could fill if they wanted.

Once the exam has started and there are some data submitted from the students, the lecturer can access the same WWW page as the exam and be given results. Results take 3 forms - listings of submitted answers per question, “Lecturer View” or “Student View”. The first lists all submissions per question, along with the original text for that question. The second, “Lecturer View” is shown immediately below and consists of a spreadsheet of raw data, reduced as much as possible. This can be exported as a tab-delimited file. The last format “Student View” is shown further below, and is exactly what the individual student sees when the marks are released. Similar reports are also generated for NEST quizzes and questionnaires.

Lecturer view: Depicted is the table of submitted data that is served to the lecturer upon access to the exam page during or after a NEST exam, quiz or questionnaire. The leftmost column is the user Id number (arbitrary), then the content of their submission as whole numbers (selection answers) words (production answers) and floating-point numbers (numeric production answers).

On the day following the examination, students are given access to their individual results. These comprise a complete tabulated report showing the student’s own answers, the expected answers and the marking scheme used to obtain their final score. Although it was possible to give students immediate access to this information, it was decided to delay this facility until the class results were viewed by the examiner.

Dotty allows the expression of more than one answer per assessment item, to cope with incomplete or inaccurate (but not incorrect) responses. In each case, the student can be awarded some sub-total of the full-mark for that item.
A student can view their personal results by following links from their on-line material (and logging in). If the lecturer wants to postpone publication, the file containing answers for the exam is moved from its proper location in the system and NEST announces that marks have not yet been released.

C. Feedback Questionnaire

Student evaluation, via a standard feedback form, is required by the student administration office for all subjects offered at the University of Melbourne. This is usually achieved by handing out “mark-sense” forms in a lecture, collecting them later on and processing them with an automatic reader. The information required by the university is a statistical summary of these data presented in a computer spreadsheet.

It was decided that such data collection was well suited to a NEST questionnaire. A questionnaire runs in a similar fashion to the NEST exam. Students merely complete the page and submit it. There is no marking. The questionnaire also has no time limit nor extra security functions, although the students do have to log into NEST.

While logging in means it is possible to associate submissions with individual students, the reporting mechanism for lecturers seen above does not supply the names of students.

Discussion and Conclusions

One limitation of NEST is the current text limit of HTML3, where there is no support for subscript, superscript and mathematical symbols in the text. This was dealt with by using “coded” representations such as brackets for subscript, eg. I(2), carets for superscript, eg. E=mc^2, and words instead of symbols, eg. kOhms. Complex equations in NEST exams have been implemented in the same way as the illustrations. After editing with a graphics package such as MacDraw, they were saved as transparent GIF files and incorporated as images in the HTML code. It is anticipated that future generations of HTML will alleviate this somewhat tedious task.

Apart from this translation problem, the interaction types supported by NEST (multi-choice, production etc) could be employed to assess questions where HTML does not allow appropriate input. Examples of these are where paper exams require students to sketch graphs. In this trial, specimen answer graphs were generated and the student was able to select the most appropriate.

In this trial exam, a total of 75 students supplied answers of typed English text, numbers, multiple choices and multiple selections. In an assessment of these submissions, the most difficult automated assessments were for free text answers because of the potential variety in what could be termed as fully or partially correct. NEST conducts searches on the first few letters of key words and allows wild card suffixes to account for normal variations on the key words. Marking of multiple choice and related types is relatively straightforward. A variant of the text answer is the numeric range answer — also marked...
with word parsing, then numeric comparison. When questions are well specified, the software has little trouble marking the students' submissions. Before marks are released, lecturers can experiment with different marking scenarios to cope with variance in production answers.

To best determine the assessment value for the exam, NEST supplies basic statistics such as variance, percentile bands and averages. Following is a graph and section of the statistics returned to the lecturer by NEST. With this facility, staff can calibrate both their assessment tools and the marking schemes used to assess submissions.

For the exam run in this trial, results were equivalent to performances on previous paper versions of the test.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>71</td>
<td>62</td>
<td>96</td>
<td>64</td>
<td>92</td>
<td>65</td>
<td>86</td>
<td>66</td>
<td>76</td>
<td>55</td>
<td>79</td>
<td>56</td>
<td>66</td>
<td>55</td>
<td>72</td>
<td>59</td>
<td>65</td>
<td>72</td>
<td>93</td>
</tr>
<tr>
<td>54</td>
<td>65</td>
<td>45</td>
<td>34</td>
<td>29</td>
<td>54</td>
<td>87</td>
<td>95</td>
<td>76</td>
<td>65</td>
<td>76</td>
<td>56</td>
<td>55</td>
<td>72</td>
<td>59</td>
<td>65</td>
<td>72</td>
<td>93</td>
<td>51</td>
<td>68</td>
</tr>
</tbody>
</table>

**Numerical results**: NEST serves statistics and a graph back to lecturers including standard deviation, per-question performance and percentile band performance for students. More complex statistics and ad-hoc queries on responses can be performed by exporting the “Lecturer View” table to a spreadsheet application like Microsoft Excel or Lotus123.

Feedback collected via the NEST questionnaire indicated that the overwhelming majority were very positive about the use of electronic media in teaching. NEST proved to be well suited to running, collating and marking electronic exams and questionnaires. Such a task is usually a tedious one for the examiner, with the redundancy that most marks end up in a computer system anyway. Additionally, marking and collating marks by hand is infamous for being error-prone, tedious and slow.

At the time of writing, NEST is being used to run two more quizzes based on previous exams. A complete user interface is under development that obviates the use of Dotty for more basic NEST documents. For more complex assessment however, Dotty would be used via a text editor in the interface.

This year, NEST will be used to run several exams within the Faculty of Engineering, and to construct a database front-end for collecting publication information from academic staff. A further application of the software is a planned survey for up to 20,000 students requested by the student union. Please direct all enquires to Craig Burton at the supplied email address or by telephone on +61 3 9344 4965.
This paper describes a Computer Managed Learning System in use at Curtin University of Technology. At Curtin 40 different subjects/units use the system and over 10,000 "student bodies" pass through the system each year.

CML an overview

The system runs from the mainframe Vax computer and provides a centralised and secure system for testing. CML:

1. Produces individualised tests

A central test bank of questions is held on the Vax from which individualised student tests can be generated at any point during the semester. You can choose as many tests as you wish during the semester.

At Curtin the testing facility (the CML lab) is centralised in the Library and is controlled by the Computing centre. A CML co-ordinator and two or three staff operate the lab. They are supported by a full time Computing staff member who also supports all staff on campus who use the system.

Therefore once CML is set up for a particular unit there is no call on computing facilities or time commitment from the staff member who is running the unit other than to maintain the data base and generating system, the "course map" and to provide face to face contact with students who have questions or concerns about their test.

2. Marks the tests

(if the questions are multiple choice, true/false or sentence completion) The system can also hold a database of paragraph or essay questions but they require a live marker.

3. Provides feedback to students on areas of weakness

Feedback is relatively limited. The students receive their results with questions to which they responded incorrectly, marked with the correct answer. Unfortunately there is no explanation available to explain why their answer was incorrect. Each incorrect item has the module and objective number listed. Students have the opportunity to check their results against their test paper and note the areas of the test in which they had difficulties.

4. Maintains a database of student results

Results are automatically logged by the system and can be downloaded at anytime into Excel.

5. Feedback on Testbank questions

CML provides feedback on how students performed on each item in the test bank and whether the question discriminated well. This allows the unit co-ordinator to modify or remove unsuitable questions.

How do you set up the system?

The syllabus needs to be divided into specific modules with objectives (outcome statement)
For example, the first module, "The cell" includes:

- **Obj 1** identify the organelles of a cell
- **Obj 2** describe the process of energy release
- **Obj 3** explain how substances pass in and out of cells

Each question in the test bank is linked to a specific objective. Questions can be further coded into cognitive levels or degree of difficulty via a header statement for each question.

For example, a question in the test bank would look like this:

The following question was 2502 (module 2 objective 5 question number 2):

- **question**
- **answer = C**
- **header = 01010101102100**
- **body**

Which of these describes the structure of the cell membrane
a) a pleated double helix
b) a micelle arrangement
c) a phospholipid bilayer
d) a crystalline lattice
e) a protein bilayer

What does the header statement mean?

- **ii**  01 instruction statement
- **tt**  01 type of question multiple choice
- **bb**  01 number of blank lines to follow question = 1
- **ll**  01 lock level - can lock questions out of the test by increasing
          lock level = 01 - is the lowest lock level and includes the item in any test drawn
- **c**   1 cognitive level - could follow Bloom’s taxonomy but can be used in any way
          wanted. An objective can be subdivided into different cognitive values
- **vv**  02 value of question - number of marks allocated
- **n**   1 number of answers expected (could have more than one)
- **dd**  00 degree of difficulty - can organise questions to make sure that all students get an
          equal range of questions of different difficulty

**How do you control CML?**

CML is controlled by a "course map" which identifies:

- the number of questions per objective
- the number of questions from each cognitive level or degree of difficulty
- the marks allocated to the question.

For example - A course map including the example question from Module 2

21/Selection = 2100 / NQ = 2/c= (1)
22/Selection = 2200 / NQ = 1/c= (1)
22/Selection = 2200 / NQ = 1/c= (2)
23/Selection = 2300 / NQ = 2/c= (1)

Would select Six questions in total, two from objective 1 cognitive level 1, one from objective 2 cognitive level 1 and one from cognitive level 2 and two questions from objective 3 cognitive level 1.

With password-controlled desktop access to the Vax the unit/subject co-ordinator can modify the test bank, change the mark allocation and cognitive levels of the questions and can modify the course map.
The day-to-day running of the system however, is in the hands of the Computing centre and the CML lab- and a nice hassle free way to run it from my experience.

**How does a student access CML?**

There are individual approaches to running CML with students. The system allows for multiple attempts on a test, practice tests before sitting the "real thing" free access to tests at any point in the semester or quite restricted access by certain dates. In our first semester unit in Human Biology we have 5 tests with specific "do by" dates whereas in second semester we have 6 tests with no time limit other than that they must complete them by the end of the teaching semester.

Students book in for a test in a very low tech way by putting their name on a list outside the CML lab. On the appointed day and time they present themselves and log onto the system to draw their test. The test is drawn from CML and is printed out by the CML staff. The CML system has logged the student’s progress and automatically produces the correct test according to the course map. The student then exchanges their photo ID student card for their test. They then take their paper and sit in the lab to complete it and the student signs the test.

Thus they do the test as a paper and pencil test.

When they have completed the test students again log onto a dummy terminal and enter their answers against the appropriate question number. When they are happy with their answers they submit them and they are marked on screen. Each student receives a print-out of the result sheet and can check it against their paper to note their incorrect answers and the correct one.

The papers are returned to the CML staff, signed and stored in case students wish to access their paper to discuss it with a staff member. Eventually all papers are shredded.

**A STUDENTS VIEW OF CML**

![Booking In the low tech booking sheet](image1)

![Getting the Test](image2)

*UniServe•Science Computer Assessment Workshop Proceedings*
DOING THE TEST

ENTERING THE ANSWERS - GETTING THE RESULTS
Advantages of using CML at Curtin

The centralised system is critical to its success at Curtin. CML provides:

1. A continuous assessment opportunity for students
2. Efficient use of your time to set up system but not have to staff it
3. Large numbers of students access to testing
4. Testing with no marking costs
5. Practice tests for students or multiple attempts at a test
6. Country campuses which are linked to the Vax direct access to the system
7. Students do not need significant computing skills — they are given instruction in accessing the dummy terminal and entering their answers.
8. At Curtin the students like it — in Human Biology they can accumulate up to 40% of their final mark and know how they're going during the semester.

Disadvantages of CML

1. This type of system tends to favour surface learning, especially if commercially available testbanks are used which often reward specific and factual detail rather than understanding — in this respect the system can only be as good as the testbank of questions accessed and that can take a lot of time and effort to develop well.
2. Question types are restricted to multiple choice, true false, sentence completion.
3. The system is text based and doesn't presently support graphics. You can get around this by using diagram books to which the student refers for a question.
4. At Curtin, country areas without access to the Vax are sent paper copies of the tests for their students - we cannot be sure that they are kept secure.

Weighing it all up

For those of us in Human Biology the CML system works well, it has high student acceptance and students often report that it helps them structure their study. The centralised system with Computing centre support works very well. One of the main success factors is the approach of the CML staff to the students. They are unfailingly kind and supportive to the students and you never hear a bad word from the students about them. This approach seems to diffuse the tension and anxiety which many students must feel. Despite the present limitations of the testbank the system works too well for us to consider any other option.

Leith Sly is the lecturer in the Computing centre who co-ordinates CML at Curtin. She is very happy to help anyone who would like the practical details of using CML. Her contact email is cslylj@cc.curtin.edu.au tel : (09) 351 2291.
Tutorial Designer

Fred Pamula
The Faculty of Science and Engineering, The Flinders University of S.A.
F.Pamula@flinders.edu.au

Introduction

Tutorial Designer is a set of programs designed to simplify the production and assessment of unique numeric tutorial papers. Tutorial Designer is used by the author to write tutorial problems using a full-featured graphical text editor. Tutorial Marker is used by the student during assessment. Tutorial Analysis is used to produce student reports that are easy to incorporate into spreadsheet programs. The following discussion outlines the process of producing tutorial papers using Tutorial Designer and their assessment using Tutorial Marker.

Tutorial Designer is suitable for numeric tutorial problems. The program was developed for the Physics Department at Flinders University but it has applications in Chemistry, Earth Sciences and Biology (Biochemistry, Genetics and Ecology). Tutorial Designer requires Windows 95 or higher to run, but Tutorial Marker and Tutorial Analysis will run on machines running Windows 3.x.

Tutorial Designer

Writing a problem in Tutorial Designer is done in stages and requires more discipline in devising problems since a numeric solution must be available for each problem. In addition, problems need to be thoroughly checked before being distributed to students. Designer has its own full featured graphical text editor so the author can write problems using any font present on the system using a full range of font attributes such as bold, italics, strike-through, underline, superscript and subscript.

Variables

Specifying a problem in Tutorial Designer is similar to the way that you normally write tutorial problems except that instead of using fixed numeric values (except of course for constants) you insert a variable. A variable is a placeholder, it can take a range of values and Tutorial Designer will allocate a value to this variable using parameters defined by the author. Variables are defined by: i) range and an increment value; ii) by entering specific values; or iii) by tying the selection of one variable with a formula that uses a previously defined variable or function.

Functions

Once a problem has been written and its associated variables defined, it is necessary to tell Tutorial Designer how to generate the correct answers. This is done by writing one or more functions that use the variables defined earlier. Functions can be chained (one function can be a composite of several other functions) so the solution to a problem can be broken into its constituent parts.

Questions

As well as defining the functions necessary to generate the correct answers, the author must write the prompts that Tutorial Marker will present to the student. At this point, you associate the formulae required to calculate the correct answer posed by the question, the accuracy of acceptable answers and the marks allocated to each question.

Tutorial Paper Header

The header that will appear on each tutorial paper is also entered in Tutorial Designer. This includes topic name and number, tutorial number, hand-up date and any special instructions that need to be given to students.
Generating files

Once problems have been written and the number of tutorial papers specified, Tutorial Designer will produce the required files so that a merge can be performed to produce the tutorial papers. Tutorial Designer produces a merge file and a table of numeric values to be used in the merge process, then an answer file and question prompts file that are used by Tutorial Marker to assess student's work. The merge file (Fig. 1) is output in Microsoft rich text format which ensures that the tutorial papers can be printed in almost any word processor and more importantly graphics inserted that help to define each problem. After loading the merge file and making any final editorial changes, a merge is performed. This process is automatic since the merge file already knows what data file to use to complete the merge process.

Tutorial papers can be saved, facilitating the reuse of previously written tutorial papers. This may take the form of editing previous problems, adding new problems or deleting problems that are no longer appropriate.

<table>
<thead>
<tr>
<th>School of Biological Sciences</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Semester I</strong></td>
</tr>
<tr>
<td>Problem Sheet 1</td>
</tr>
<tr>
<td>Due March 31, 1996</td>
</tr>
</tbody>
</table>

*Please enter your answers into the Tutorial Marker program in the Computer Mediated Learning Unit.*

Problem 1
How many mols are there in «p1v1» g of a compound with a molecular weight of «p1v2» g.mol\(^{-1}\)?

Problem 2
What is the concentration in mM of a «p2v1» ml solution if «p2v2» mg of a compound with a molecular weight of «p2v3» g.mol\(^{-1}\)?

Problem 3
How many mg needs to be weighed out if the molecular weight is «p3v1» g.mol\(^{-1}\) to make a «p3v3» ml solution of «p3v2» mM?

Problem 4
What is the molar concentration if the absorbance was «p4v1» and the extinction coefficient is «p4v2» l.mol\(^{-1}\).cm\(^{-1}\) and the path length is «p4v3» ?

**Figure 1.** A typical merge file produced by Tutorial Designer prior to performing a merge. During the merge the fields marked <<pxvy>> (px represents problem x and vy, variable y) are substituted with values generated by Tutorial Designer and for which answers have been calculated.

Tutorial Marker

*Introduction*

Tutorial Marker is used by students to have their work marked. When a student registers, the program retrieves the student's record and then guides the student through the tutorial paper.

*Registration*

Registration requires the student be identified so that the program can keep track of the student’s results. Students enter their surname, first name, student number and password into the form (Fig. 2). The Surname and First Name fields accept a maximum of 19 characters while the Student Number and Password fields accept a maximum of 9 characters. The program provides password protection by not displaying the student’s password but a series of asterisks (******). An authentication process is employed which ensures that only those students enrolled in a course have access to the software for that course though this can be disabled.
Marking

Once students have registered, they are free to enter answers to their tutorial paper. Students can enter the number in scientific notation if required and then click the Mark button to have their work marked. The student is then informed if the answer is correct or not.

Figure 3. Marking form window. An answer is marked by clicking the mark button. If a question has previously been answered correctly, the correct answer will appear in the edit window.

Score Sheet

Students can look at their accumulated score by clicking the Score button. A list of problems and the marks scored for each is shown (Fig. 4). In addition to the student's score, the number of attempts at each question, the number of logon attempts and the total amount of time working on the program are also recorded.
Post tutorial

After the submission deadline, the answer file is unencrypted, formatted and sorted by Tutorial Analysis. Finally, Tutorial Marker can be set so that it displays the correct answers but does not allow students to alter their record after the submission deadline. This ensures that students can see the correct numeric values for their tutorial paper.

Computer Based Assessment

Tutorial Designer reflects very closely the way we traditionally handle tutorial papers. Tutorial papers are distributed to students and students are expected to work on these at home and submit them for assessment. Using Tutorial Designer, assessment is handled electronically. Electronic marking makes some demands that we do not normally need to deal with. Students must be given access to the computer based assessment system. This requires that a password be assigned to each student. Passwords can be issued to students or left up to the student to generate. Perhaps the best approach is to issue passwords at the beginning of the year and then make students change them at their first logon attempt. Although not essential, it is desirable to have some authentication process which ensures students have access to those programs required by the courses in which the student is enrolled.

The approach to setting up a file system is also important to consider. A system can be constructed so that all student records are placed in one file or alternatively, each student has a single file. Irrespective of the approach adopted, it is important that student results are written to disk in a way that does not overload the server.

Finally, it is important to ensure that the student records cannot be tampered with. This usually means that when students logon to a system they do not have access to any software tools that would allow them to either see or edit their or other student records. Failure to ensure this could result in students losing respect for the system and refusing to use computer based assessment.

Summary

Setting up effective computer based assessment requires the resolution of a number of issues including security, access to computer terminals and the appropriateness of computer based assessment to the course. The approach finally taken dictates how students will respond to the exercise. One program developed by the Faculty of Science and Engineering at Flinders University called Tutorial Designer was presented to illustrate many of these issues.
Introduction

The computer assessment "ralph" is briefly described. The design features are enumerated to establish what ralph does and does not do. Details of the operation from the point of view first, of the student, then, of the tutor, follow. Finally, some practical issues are addressed and an overall evaluation is given.

Design Features of RALPH

The software package "ralph" is designed strictly for CML - computer managed learning - in contrast to CAI - computer aided instruction. The computer manages learning, rather than delivers content. To put this concept in human terms, the computer is replacing a manager, or clerk, rather than an instructor. It is important that this distinction be appreciated at the outset.

More specifically, the operation is described by the acronym CMA - computer managed assignments. Hence ralph is right at home at this workshop dealing with computer assessment. Simply put, ralph issues, collects and marks assignments. Hence ralph is right at home at this workshop dealing with computer assessment. Simply put, ralph issues, collects and marks assignments.

The environment in which ralph operates is well-described as "terminal-mainframe", or "client-server". "Connectivity" is intrinsic to ralph. Ralph is not designed to operate on a stand-alone PC. The code itself and the data reside on a central computer. Perhaps five years ago, when putting a PC on every desk was a goal, this might have been seen as a drawback; but now, with the growth of the global network, it is seen as an asset. As network connections grow, so do the tentacles of ralph. Ralph is typically run from terminals on campus, which are found in the physics laboratory, the library, and terminal rooms; and by students dialling in from home, work, or student residences. Ralph can be run from anywhere in the world where there is telnet access. This allows extremely "flexible delivery". As a specific example, the workshop demonstration proceeded by invoking ralph on a computer at Wollongong from terminal rooms at Sydney University.

Everything about ralph is designed to be lean. The code is fast. The file sizes are small. Student time at the terminal is kept to a minimum. CPU time is kept down. These issues become particularly important as class size increases. Ralph is typically run in classes of hundreds of students, and thousands of students, answering tens of thousands of assignments, are easily handled by one machine.

Ralph is designed to be a very general computer assignment system. While originally developed for the physics department, question banks can be generated for any discipline. Ralph supports a variety of question types: multiple choice, numerical (including units) and keyword. The file format for question banks is simple ASCII. Likewise, simple ASCII output is available on line printers, or dumb terminals; for formatted output, the typesetting system TeX is employed. The use of these very general file types has encouraged use of ralph on various platforms, e.g. PC, Macintosh and UNIX.

Student Operation

For the student, using ralph is as simple as 1, 2, 3: 1. get an assignment, 2. solve the problems, 3. submit the answers. Steps 1 and 3 are performed at the computer terminal and typically take a matter of seconds and minutes, respectively. Step 2 is performed away from the terminal and is where the learning takes place, namely, through problem-solving. Solving the problems typically takes of the order of an hour.
Once the student has logged on to the host computer, any special "message" that the tutor wishes to bring to the attention of all students is displayed. The menu of student commands is then as follows:

<table>
<thead>
<tr>
<th>NAME</th>
<th>SHORT</th>
<th>OPTIONS</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>g</td>
<td>[number]</td>
<td>draw an assignment</td>
</tr>
<tr>
<td>print</td>
<td>p</td>
<td>[number]</td>
<td>re-print an assignment</td>
</tr>
<tr>
<td>submit</td>
<td>s</td>
<td></td>
<td>answer the assignment</td>
</tr>
<tr>
<td>verify</td>
<td>v</td>
<td>[n/all]</td>
<td>check assignment marks, number can be &quot;all&quot; for all assignments check assignment marks</td>
</tr>
<tr>
<td>help</td>
<td>h</td>
<td>[command]</td>
<td>display help for a command</td>
</tr>
<tr>
<td>units</td>
<td>u</td>
<td></td>
<td>display the table of units</td>
</tr>
<tr>
<td>password</td>
<td>pw</td>
<td></td>
<td>change one's Ralph password</td>
</tr>
<tr>
<td>quit</td>
<td>q</td>
<td></td>
<td>exit</td>
</tr>
</tbody>
</table>

These are arranged in the normal chronological order in which the student would employ them. First, the student "gets" an assignment. This is then printed, typically on a laser printer. It may alternatively be printed on a line printer, displayed on the terminal, or dumped for file capture. The student is then logged off the system. If necessary, the student may at any time "print" another copy of the assignment.

After attempting the problems, the student returns to the computer to submit the answers. The command to allow this is "submit". The computer prompts for each answer, indicating the type of answer required, e.g. floating point number, unit name, keyword, or multiple choice letter. The student is given the chance to cancel the submission at any time. Once the answers are submitted, they are marked, with the results given (practically) instantly. For each answer the student receives a mark and a single line of feedback, which is to provide assistance in the case of a wrong answer. At any time the student can check the record of assignments received and submitted using the "verify" command.

"Help" is available at any time by typing that word; "units" brings up a list of units that ralph recognises (e.g. for time, second, minute, hour; but not week). To deal with a specific arrangement when all students use the same login name, ralph runs its own user identification system, and hence the command "password".

**Tutor Operation**

The tutor is the person responsible for the academic side of ralph, typically an academic staff member. The main roles of the tutor are to produce and maintain a question bank and to monitor student progress. There are various administrative tasks which are often performed by the tutor, such as adding or deleting student users; and submitting student answers, over-riding normal system checks when, for example, a student wishes to submit an assignment after the deadline, due to illness. The main commands available to the tutor are given in this menu:
<table>
<thead>
<tr>
<th>NAME</th>
<th>SHORT ARGUMENT</th>
<th>Brief comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>help</td>
<td>h [name]</td>
<td>display help messages</td>
</tr>
<tr>
<td>* question</td>
<td>qn</td>
<td>enter question bank administration</td>
</tr>
<tr>
<td>* account</td>
<td>ac</td>
<td>enter accounting mode of the system</td>
</tr>
<tr>
<td>* system</td>
<td>sys</td>
<td>enter system administration</td>
</tr>
<tr>
<td>loadu</td>
<td>lu [name]</td>
<td>load user accounts from file `name'</td>
</tr>
<tr>
<td>editmsg</td>
<td>e [editor]</td>
<td>edit the `message of the day' file</td>
</tr>
<tr>
<td>user</td>
<td>u</td>
<td>add a new user to the system</td>
</tr>
<tr>
<td>duser</td>
<td>du [name]</td>
<td>delete a user with login &quot;name&quot;</td>
</tr>
<tr>
<td>listu</td>
<td>[n][name]</td>
<td>list users, n is option to sort by name</td>
</tr>
<tr>
<td>symbols</td>
<td>sym</td>
<td>display the table of symbols</td>
</tr>
<tr>
<td>dass</td>
<td>da</td>
<td>delete an attempt on an assignment</td>
</tr>
<tr>
<td>password</td>
<td>name</td>
<td>change the password for user &quot;name&quot;</td>
</tr>
<tr>
<td>pstat</td>
<td>p</td>
<td>show printing log on all printer queues</td>
</tr>
<tr>
<td>quit</td>
<td>q</td>
<td>get back to previous level</td>
</tr>
</tbody>
</table>

As for the student, additional "help" is available for each command. The three main commands have more detailed submenus, which will not be shown here, but each of these areas will be discussed in turn.

The "question" command allows access to the question bank. Prepared question banks may be "compiled" to ensure that they are formatted correctly, then "loaded"; the complete question bank may also be "printed". Individual questions may be "deleted" (typically, if a mistake is detected after loading) and "undeleted". The "figure" command allows figures to be added. Finally, a suite of commands allow the tutor to construct and modify assignments. The tutor specifies release and expiry date and number of questions, and is presented with a brief list of the available questions from which to choose. Questions may be of multiple-choice, keyword, or numerical type. Either "exact" or "approximate" match (for which part-marks are given) for keyword answers is permitted. A tolerance may be specified for numerical answers, as well as the proportion of marks to be allocated for the units.

The "accounting" submenu provides a variety of ways of viewing the students' marks. Any combination of assignments may be chosen, and the amount of detail may vary from the individual marks for each attempt at each assignment for each student, through to a histogram showing the performance of the class as a whole; at the end of the session the option that gives the average mark against each student name is used. There are also "timing" diagnostics, which show how often a student logs in, how long the student takes between drawing the assignment and submitting, and so forth, and "question" diagnostics, which indicate how often the question is answered correctly.

The "system" commands are used to add or delete courses, examine the student data record in details, lock the system from public access, check the system load, and so on. Some of these commands are available only to the "superuser", generally a computer operator, who sets up ralph on the computer and then enrolls tutors. The "superuser" category of user will not be discussed further here.

**Running RALPH**

Several practical issues need to be considered in setting up and running a computer assessment system of this nature. As far as hardware is concerned, ralph needs a UNIX box. The preferred operating system is Solaris 2.5, but ralph has been run on many computers running many flavours of UNIX (e.g., Linux on a PC, Ultrix on DEC, Dynix on Sequent, SunOS(BSD) on Sun Sparc). This box then needs to be connected to a network to which telnet connection can be made. Printers, ideally laser printers, need to be available on which to print the assignments.

As far as personnel are concerned, a computer support person (with UNIX background) is needed to install and oversee the operation of ralph. For each course that is being run a responsible academic
needs to be appointed. Regular, marked assignments tend to generate among the student body a lot more interest in the subject matter than otherwise, and academic support to answer student queries must be available.

**Evaluation of RALPH**

At the outset of this development it was recognised that technological issues were subsidiary to, and should not drive, the academic goals. In view of the sometimes conflicting and sometimes fantastic reports regarding the efficacy of computer assessment systems, a careful analysis was undertaken on the effect on examination marks of a student using ralph, compared to the student attending the alternative problem solving class. A small, but significant, increase in marks was found for the student using ralph. The details of this analysis are given in the first paper listed in the bibliography.

A number of student surveys have been conducted to determine, qualitatively and quantitatively, what the students think of the system. They are generally positive. For example, in a survey last year from first year chemistry students at UNSW, the final summative question asked "would you like to see this system used again?"; 79% replied "yes" and 21% "no". What students repeatedly say they like about ralph is that (a) it is flexible; they can log in and use it at any time; (b) it encourages work, a discipline that, in the absence of regular, marked assignments, they may not be able to self-impose; and (c) the marking, done by a computer, is fast, and perceived to be objective.

**References**


