A CASE STUDY IN ELECTRONICALLY ASSISTED PEER ASSESSMENT

Ashley Ward (ashley@dcs.warwick.ac.uk)
February 2000

Design of Information Structures (‘DIS’: CS126) is a module which aspires to give a basic grounding in some fundamentals of Computer Science (the module aims are detailed in Appendix ii). It is a core module for students studying courses with a high Computer Science content, and is also available as an option for Mathematicians and others who have studied the prerequisite module CS118. Currently, around two hundred students in the second term of their first year (and hence with widely varied experience) each attend thirty hours of lectures and eight hours of lab sessions.

I first became involved with DIS in 1999. Dr. Abhir Bhalerao (a lecturer on the module, and incidentally a 1998/1999 WTC[1] participant) was in the process of redesigning the lecture support for the module, replacing little-attended seminars with compulsory lab sessions. Worksheets were drawn up for four lab sessions, each presenting the learner with problems to be solved and supporting explanatory material (an example is provided in Appendix vii). Each lab session ended with a thirty-minute test run under exam conditions, augmenting the existing assessment methods (a programming assignment and a more traditional examination) with a small amount of credit. This credit seemed to ensure a high attendance figure, but the marking of the paper-based tests (see Appendix ix for an example) created problems. With several reams of paper representing the 800 scripts to be marked by the four unlucky demonstrators, just distributing the scripts for

[1] The assumed audience for this document is a participant on the Warwick Teaching Certificate course.

marking, then collating and publishing the results (a task I took on) was a very time consuming activity. The marking process was similarly unrewarding, and we quickly fell behind. Eventually a simple grading (from 0-3) of each script was all that was achieved. Each script was marked only once, no moderation was performed, and the results took several weeks to arrive. The whole process was quite unsatisfactory - a large amount of effort had resulted in very bored demonstrators and only minimally useful feedback (see the feedback sheet, Appendix viii) for the students.

The need for automation seemed obvious, but I wanted to improve the quality of the feedback. One of the aims of the module (see Appendix ii) is “to learn how to program…”. Some of the questions in the 1999 tests could certainly be rewritten as ‘closed’ multiple choice questions (MCQs) which are easily marked automatically. However, if our assessment is to match our objectives, we must assess programs (or at least code snippets) that have been *constructed by the learners*. Unfortunately, it is difficult to build an automated process that provides meaningful feedback about the construction of a program - the problem would be similar to that of conceiving an automatic system to give feedback about this essay. Simple measurements (e.g. the number of comments in the code) can be made, but an automated process cannot perceive the more subtle aspects: choice of variable names; elegance of solution; *why* this code would not give the required result. Without a highly constrained specification, an automatic process cannot even perform a simple test for correctness [Joy]. In short, removal of the human element from a code assessment process reduces the quality of assessment. Automated marking in the DIS tests would not improve the quality of feedback, and if we desire to improve this, we are still left with a large number of scripts requiring considered human inspection.

Although computers cannot make effective judgements of scripts, they can certainly simplify the document management problems. Could a system facilitate the passing of information from learners to assessors and collate the marks? Dr. Mike Joy (another previous WTC participant) has constructed such a system named BOSS [Joy], and it handles the management of the DIS programming assignment well. Unfortunately, the actual marking, not the document management, is the larger portion of our tests problem, given that 800 scripts require timely assessment.

Stalemate ensued, until the WTC coursework self-assessment forms, and later the presentation of "'control' and 'independence' strategies" [Gibbs, p.163], inspired me to consider a different perspective, which I will now outline. Given an imaginary 200 assessors, say, the total amount of time available for marking would be much greater than with just our four demonstrators (assuming a similar boredom threshold). Each assessor could spend a much longer time with each script than previously, providing a good amount of specific feedback. Indeed, each script could easily be marked multiple times. If assisted by appropriate document management, all this marking could be done simultaneously, producing detailed feedback in a very short time. But, in practice, where could 200 assessors be found? The easiest solution is to use the group of learners currently engaged in study on the module - ergo we need peer assessment (unless 200 relatively more experienced second years or members of staff can be coerced into the job).

Peer assessment immediately raises the spectre of “the blind leading the blind” - how can learners help each other when they do not fully understand the material[2]? I believe there to be no absolute solution to this problem, but two techniques can be used which should minimise this possibility. If each assessor marks, say, three scripts, then each script can be marked three times. If each script is given three marks, then the variance of the marks can be calculated, and if unusually high (indicating disagreement between the assessors), the script highlighted for moderation by an expert. The second technique involves some clever distribution of scripts for marking. The distribution algorithm could attempt to assign scripts such that each script gets marked by assessors who are perceived to be spread across the ability scale (one ‘expert’ assessor, one average, and one below average, perhaps). The perceived ability could be computed from automatically marked MCQs, or from previously gained marks.

These worries surrounding peer assessment can also be countered by pointing to the long list of advantages of a system with 200 assessors. The most obvious is the possibility for fast and human feedback, in triplicate. Asking learners to evaluate work is also a justifiable objective: they are asked to read as well as write code (a important skill when the reality in industry is that systems are only very rarely built entirely from scratch). It has also been said that “the best way to learn is to teach” - and here, in the writing of feedback, is an opportunity for learners to teach. More abstractly, evaluation is an active process, encouraging reflection upon and discussion of one’s own answers - all hopefully fostering the ‘deep’ (rather than ‘surface’) approach to learning examined in [Gibbs,

[2] Or, as an email from a student involved in the process put it, “Please get someone with intelligence to

p.155]. Seeing the multiple opinions expressed by other assessors about one piece of work may also provide experience that helps learners to progress from Perry's 'dualist' caricature through 'multiplism' and 'relativism' to 'commitment' [Perry, p.146]. Finally, such a system would allow tutors to concentrate more of their efforts on teaching and moderation.

Several on-line assessment systems are available, including [MERLIN] at Hull, [CASTLE] at Leicester, [COSE] at Staffordshire and [TRIADS] at Derby. However, none of these appear to have a peer assessment element, and so we started to design a system from scratch: an *O*n-line *A*ssessment *SYS*tem (OASYS).

During this phase, I examined several considerations, the first being that of anonymity. Anonymous marking is a University requirement if the work is for course credit, and in a peer assessment system, it becomes an Equal Opportunities issue: anonymity is a crucial element to include if we wish a group of learners with a wide range of starting abilities to take part.

My experience with the tests in the previous year also raised the issue of flexibility: at the design stage, it was difficult to tell which students would attend the module, and I anticipated that significant numbers would not attend the correct lab sessions at the correct time. To solve the first problem, I built a self-registration system. This operates as follows: the student enters their ITS ("Information Technology Services") username and library card number into a registration page. If the two match (this information is

mark the tests"!

publicly available), then the system sends an email containing a generated password (we cannot use their ITS password due to the lower level of security in this system) to their ITS email account. This therefore guarantees correct authorisation, as long as the University regulations relating to ITS computer accounts being for personal use hold.

In an attempt to deal with the second mentioned problem regarding attendance, I decided to make the system as self-managing as possible by calculating constraints in real-time from data representing user actions, not from a preconceived timetable. The task of checking these constraints was made difficult by the complexity of the published lab timetable (see Appendix ii), which is skewed due to a small number of demonstrators and rooms, resulting in a total of 96 person hours of demonstrator time for the lab sessions. It was hard to guarantee that the logistical arrangement was actually possible – would enough scripts be available for marking in time, how long would it take feedback to arrive, and other questions. I designed the constraints (the primary result being "learners must mark three scripts by the start of their next session") with the aid of a spreadsheet (Appendix vi) and a small simulation model that I constructed. There is a trade-off in the design between feedback time and the possibility of cheating (since when marking, learners see the correct answers), but I chose in favour of speedy feedback.

The resulting implementation is described in the "OASYS tour" (Appendix i), which I will not re-describe in depth here. There are currently three possible question types: free response questions; MCQs, which were introduced primarily to allow the clever script distribution mechanism mentioned above (which unfortunately is not currently implemented), and Permutational Multiple Choice Questions (PMCQs), which are justified in [Farthing] and which “seek to assess high level learning and overcome the problem of candidates guessing”. The questions are designed in Hyper Text Markup Language (HTML) and hence are quite independent of DIS – there are opportunities for using this system on other modules, both in the Computer Science department and beyond. The initial system (consisting of registration, testing and marking) was conceived, designed and implemented (including the choice and installation of web server, scripting language and database) in only two weeks – the rest of the system has been constructed during operation. I am responsible for the majority of the system: the back-end, the student-facing interfaces, and most of the supporting documentation. My colleague Abhir constructed the questions and the majority of the administration interfaces, including the question editing interface.

Table 1 compares OASYS to the 1999 paper-based system.

Table 1: On-line assessment system as compared with paper-based assessment system

Advantages	**Disadvantages**
• Instant feedback for learners. • Easy analysis of scripts, perhaps in real-time: e.g. it is possible to see a histogram of answers submitted during a test and hence determine when the majority of the class has finished. • Concurrent access to scripts. • It is not so easy to lose a script!	• Forgotten passwords *et cetera* create problems during the lab sessions. However, there is no possibility of a script of uncertain origin. • Mistakes are harder to rectify (for the Computer Scientists – paper-based assessment has high availability and redundancy built in!) • Work on setting up the tests cannot be delayed – the entire test, hints for marking and a mark scheme must all be ready by the time the test begins. • Special cases are hard to cater for – every restriction put into the system is bound to encounter an exception at some stage.

Having described the problem and our solution, I will move on to examine our work from an educationally theoretical perspective. The decision that labs should replace the badly attended seminars seems well justified. Lectures may be a cost effective way to transmit information representing the theory of DIS [Bligh, p.10], but at least two of the three modules' aims refer to practical skills: "…gain familiarity with specification, implementation and use of … ADTs…", "…learn how to program … using an object-oriented language" (see Appendix ii). The practical skill of computer programming is acquired only through experience: it is necessary to make some transactions with the environment [Kolb, p.36] (in programming, this would be the compiler, the run-time system, and, in a commercial context, eventually the users of your product) in order to validate our many assumptions about the syntax and semantics of our code. Often, we will discover that our assumptions were incorrect - this is a "useful mistake" [Neill].

Hegel, quoted in [Kolb, p.28] puts it more strongly: “any experience that does not violate expectation is not worthy of the name experience”. This is especially important for those students that already have programming experience, but with a language other than the one used in DIS. To use a current example, when taking seminars during term one this year, I noticed that students with experience in Visual Basic had particular problems with the Java `for` loop construct. Certainly, “all learning is re-learning”, or, put another way, “learning is a continuous process grounded in experience” [Kolb, p.28]. Practical lab sessions are an excellent way of cultivating the experiences that I have outlined, and so they are easily justified as an essential part of the DIS module.

Now, to move our evaluative eye to the tests within the lab session. There are two objectives behind our assessment through testing. Firstly, the credit available is a carrot used to encourage student attendance at these experience-generating sessions. Secondly, it assesses learners’ abilities in the area of practical coding, in a situated and hence more valid (“…measures the intended aims…” [Cotton, p.92]) way than assessment during traditional examination. Unfortunately, these aims are not crystal clear in the introduction to the testing system that I wrote (Appendix iii) - clearer statements of learning outcomes (a term first used by Otter and clarified in [Allan]) should be used next year. Another unfortunate lack of clarity occurs in the marking criteria document (Appendix iv). It is now obvious to me (after reading [Cotton]) that the marking should be criterion-referenced: “somebody decides the standards which have to be achieved before an award can be given to an individual” [Cotton, p.39]. We conflate several assessors ordinal scales [Cotton, p.73] in the system in order to generate overall marks (although not in the summary sheet - see Appendix i). Clear statements of standards to be achieved for each grading would improve the reliability ("…achieve the same … result whenever the method is used…" [Cotton, p.92]) of these results. Finally, an experience with a stressed and tearful student (who admitted being a perfectionist) recently highlighted to me the teacher-centred nature of our testing: there is an argument "that the whole notion of autonomous learning is undermined if assessment is determined unilaterally by staff" [Boud quoting, p.36]. The imposition of a 30 minute time scale on the test certainly denies individuals freedom to work at their own pace, although this is offset a little by the provision for individuals to mark at their own pace, in their own time. It is hard to see how the tests could be made more flexible but still retain the aforementioned carrot factor, thus encouraging attendance.

Whether this experiment has been successful however remains to be seen. A questionnaire could assess the popularity of the system with the students, and we need to undertake a quality assessment of the marking performed by the learners, probably during the moderation process. To conclude, there certainly appears to be a bilateral form of dependence between electronic and peer assessment. We cannot have good (i.e. speedy and anonymous) peer assessment at this scale without electronic assistance, and conversely, we cannot have good marking (i.e. quality feedback) of scripts without humans.

REFERENCES

[Allan] Allan, J. (1996). Learning Outcomes in Higher Education, *Studies in Higher Education* 21, 1, 93-108.

[Bligh] Bligh, D. (1998) *What's the Use of Lectures?* Exeter: Intellect.

[Boud] Boud, D. (1995) *Developing Student Autonomy in Learning*. Norwich: ERTEC/UEA.

[CASTLE] `http://www.le.ac.uk/castle/`

[COSE] `http://www.staffs.ac.uk/COSE/`

[Cotton] Cotton, J. (1995) *The Theory of Assessment.* London: Kogan Page

[Farthing] Farthing, Dave W. (University of Glamorgan, 1998) Best Practice in Non-Subjective Assessment. *Monitor* (Journal of the CTI Centre for Computing) Summer 1998, pp.24-26.

[Gibbs] Gibbs, G. (1995) Improving the Quality of Student Learning Through Course Design, in Barnett, R. *Learning to Effect*. Buckingham: SRHE/OUP, 148-165.

[Joy] Joy, M. & Luck, M. (University of Warwick, 1998) The BOSS system for On-line Submission and Assessment. *Monitor* (Journal of the CTI Centre for Computing) Summer 1998, pp.27-29.

[Kolb] Kolb, D. A. (1984) *Experiential Learning: Experience as the Source of Learning and Development*. New Jersey: Prentice Hall.

[MERLIN] `http://www.hull.ac.uk/merlin/welcome.html`

[Neill] Neill, Sean. "Useful mistake" was a term contributed by Sean during a research seminar I attended in 1999.

[Perry] Perry, G. (1988) Different Worlds in the Same Classroom, in Ramsden, P. (ed) *Improving Learning: New Perspectives*. London: Kogan Page, 145-161.

[TRIADS] Mackenzie, Don *Recent Developments in the Tripartite Interactive Assessment Delivery System (TRIADS).* University of Derby.

APPENDIX

i. “OASYS tour”, showing and explaining the system from the learner and administrator points of view.
ii. DIS home page showing module aims, lab timetable…
iii. OASYS: Introduction page, which attempts to outline aims and objectives for the system.
iv. OASYS: Criteria explanations page, giving general guidelines for marking.
v. OASYS: Marking calculations FAQ (Frequently Asked Questions) page, answering common questions about learner’s marks.
vi. Logistics spreadsheet – will there be scripts available for marking at the right time? How long will the feedback process take?
vii. Lab session 4 worksheet, which learners work through before attempting the test (print out from on-line).
viii. 1999 feedback sheet – minimalist and not useful for the learners.
ix. 1999 assessment test 4, showing the equivalent paper-based testing.