Enhancing the Learning Experience on Programming-focused Courses via Electronic Assessment Tools

Hans-Wolfgang Loidl\textsuperscript{1} Phil Barker\textsuperscript{1} Sanusi S. Usman\textsuperscript{1}

\textsuperscript{1}School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh

TFPIE17: Trends in Functional Programming in Education

\textsuperscript{*}The Scottish Informatics & Computer Science Alliance
A Pedagogic Case for the Use of E-exams

- Improve *feedback*: use formative e-exams to provide immediate feedback to students and provide a *checkpoint* of their progress
- Improve *student engagement*: exams and tests are peak times of student engagement;
- *Programming courses are ill-suited for paper-based exams*
- Counter-balance classic *coursework* (for deep learning) with regular *e-class-tests* for basic knowledge checks (“threshold knowledge”)
- E-exams help matching questions/tasks to learning objectives, and allow to develop a portfolio of questions

0https://blogs.pjjk.net/phil/requirements-online-exam-system/
Wider Rationale for E-exams

- Deal with *large first-year programming courses* and reduce the amount of (manual) marking.
- Use a model that *scales* to several campuses (Edinburgh, Dubai; Malaysia?)
- Support *supervised distance learning* as well (Graduate Level Apprenticeships; new!)
- Analyse the exam results to detect trends
- Give *immediate feedback* to students on tests and quizzes.
- *Increasing student engagement* and providing checkpoints of learning progress for the students;
Core requirements for the exam software are:

- **security**: providing a lock-down client
- **reliability**: tested in various settings and on a large scale
- **scalability**: caters for large classes and different campuses
A partial solution: BTL Surpass

- The flagship product by BTL for both **summative and formative exams** (http://www.btl.com/surpass/)
- Used on large classes (hundreds of students)
- Used across several learning domains
- Good support of the workflow from course design to statistical evaluation of the results
- Already used at Heriot-Watt for overseas exams (in Business)
- A range of question styles, from multiple-choice to fill-in-the-blank
- Provides a **secure, lock-down client**
Courses

“Software Development 1” (Year 1, Semester 1):
introductory Java programming;
  ▶ large class size;
  ▶ mainly summative assessment;
  ▶ high stakes exam

“Hardware-Sofware Interface” (Year 2, Semester 2):
systems programming in C and ARM Assembler;
  ▶ challenging for students;
  ▶ sizable coursework is daunting to students;
  ▶ ⇒ provide checkpoints (formative tests, mock exam)

“Distributed and Parallel Programming” (Year 4,
Semester 2): a range of different parallel programming
technologies;
  ▶ students struggle with the number of technologies;
  ▶ appreciation of relative ease of programming in a
    high-level language;
  ▶ provides a step-change in learning prg langs
Using e-exams in software development courses

Results from using BTL Surpass in a *summative exams* in “Software Development 1”¹:

| I was able to demonstrate whether I understood the material being tested. | 1  2 12 25  19 |
|---|---|---|---|---|---|
| I felt adequately prepared to use the software. | 1  4  8 24  23 |
| I found the software used to deliver the exam easy to use. | 0  2  7 14  37 |
| I was able to enter the answers that I wanted to. | 1  1  6 12  40 |
| The feedback from the software about whether my answers had been submitted was adequate. | 0  5 12 20  23 |
| The computer lab was a suitable setting for this exam | 1  4 10 13  32 |

¹results from 60 out of 150 students; on a 1–5 likert scale
More results and Student responses

- Highest strongly agree (42 strongly agree, 7 agree, out of 50) was on “I would like to be able to self-assess for practice while studying”

- with a desire expressed for “immediate automatic feedback on my performance” (44 strongly agree, 10 agree).

- From interviews
  - easier and less stress-ful to work with an on-line system rather than paper-based exams
  - useful to flag questions during the exam
  - useful to have a timer on screen
Results on Feedback

These results are from a **mock exam on “Hardware Software Interface”** involving 12 students.

Self-assessment: *I would like to be able to self-assess for practice while studying*

---

**Note:** results on a reverse 1–5 likert scale, i.e. 1=strong agree

---

1. Pictures from https://openclipart.org/
**Results on Feedback**

*Immediate feedback:* I find immediate automatic feedback on my performance really useful.

---

1^ Note: results on a reverse 1–5 likert scale, i.e. 1=strong agree

1^Pictures from https://openclipart.org/
Results on Usability

Understanding: *I was able to demonstrate whether I understood the material being tested.*

Note: results on a reverse 1–5 likert scale, i.e. 1=strong agree

Pictures from https://openclipart.org/
Results on Usability

Ease-of-use: I found the software used to deliver the exam easy to use.

\[1\text{ Note: results on a reverse 1–5 likert scale, i.e. } 1=\text{strong agree}\]

\[1\text{Pictures from https://openclipart.org/}\]
Results on Usability

Lab: *The computer lab was a suitable setting for this exam.*

1\(^{\text{Note}}\): results on a *reverse* 1–5 likert scale, i.e. 1=strong agree

\(^{1}\)Pictures from https://openclipart.org/
Results on Usability

Overall satisfaction: *Overall, were you happy with the software that was used during your previous online assessment?*

![Bar graph](https://openclipart.org/)

1. **Note**: results on a *reverse* 1–5 likert scale, i.e. 1=strong agree
1. **Pictures from https://openclipart.org/**
Conclusions

- Coding skills can be effectively tested in an e-exam setting
- Students like immediate feedback (on quizzes and e-exams)
- Mixed from results second year course: very positive on issues of “immediate feedback”, but negative on some usability issues
- Potential for usage in fourth year: the step-change in learning a functional programming paradigm needs a mixture of lecture-style presentations (use e-tests for “threshold knowledge”) with practical coding exercise (MOOC style)
Some Practical Lessons

- Don’t make the a “high-stake exam” the first meeting point of student and software
- Manage the student’s expectations
- Pay attention to logistics around setting up the exam: login data, lab setup (and space), lab helpers
- Know the system well enough to deal with hiccups
- Make sure to have on-site technical support
Related Work

- **Coderunner:** run and test code submitted by students; supports many different languages; implemented as a Moodle plug-in
- **Coding Bat:** intended only for small scale coding practice, but might still be useful
- **STACK:** computer aided assessment of mathematics, using an open-source computer algebra system underneath; implemented as a Moodle plug-in
- **The Great Courses:** this is a commercial site that provides video lectures for a lay audience; notable for the high quality of lecture presentations