E-exams and exam process improvement

Guttorm Sindre, Aparna Vegendla
Department of Computer and Information Science (IDI), NTNU
{fornavn.etternavn@idi.ntnu.no}

Abstract

While many tasks concerning the interaction between students and learning institutions have been successfully digitized, high-stakes examinations remain mainly a traditional paper-based process in most Norwegian learning institutions. A large scale shift towards e-exams can however be expected during the next 5-10 years, based on a number of perceived benefits of digital exams over paper-based exams. From a pedagogical perspective it is important to avoid a too narrow focus on e-exams mainly as a means to save money by making the examination and grading process more efficient, but rather as a possibility for more fundamental process and quality improvement. This paper analyzes what improvements might be possible, and what requirements this would entail for the e-exam system.

1. Introduction

There has been a long term interest in providing various types of computer support for university examinations. A survey by Brusilovsky and Miller [1], specifically focusing on distance education, showed a lot of technologies available for web-based testing of students already in 1999. Technologies were analyzed with respect to three stages, preparation (e.g., the teacher developing the test questions), delivery (making questions available to students and collecting their responses), and assessment, such as grading and providing feedback. Sclater and Howie [2] presented requirements for what they considered the "ultimate" assessment engine in 2003, and analyzed two existing commercial products with respect to these requirements. Their finding was that none of the products fully satisfied the requirements, but most of them were actually satisfied quite well, showing that the packaged solutions were viable. Kuikka et al. [3] list some wanted features for e-exam systems, and compares some existing LMS tools for what they offer. A number of advantages have been reported for computer-based assessments over traditional paper-based examinations [2-9]:

- *for question and test development:* automated question generation from templates, sharing of questions between learning institutions through question banks, more complex item types, including multimedia and interactive material, as well as adaptive questions.
- *for delivery of the exam:* reduced cost due to avoiding paper, both in distributing questions and collecting answers. Computer support might also make it easier to adapt the exam to students with special needs, or students located far away from the normal exam venue. Most students also prefer typewriting to handwriting, as they are nowadays little accustomed to handwriting for learning activities outside the exam room.
- *for assessment:* reduced cost of distributing answers to graders, possible automated or semi-automated support for grading (depending on question types), easier to read typed than handwritten text. Also anonymity may be better with typed text, as handwriting may sometimes give away candidate identity in spite of anonymous labeling schemes. Formative feedback and the handling of appeals would also be easier with the answers and grader comments and marks available electronically.

This paper was presented at the UDIT / NIK 2015 conference. For more info see http://www.nik.no/

In spite of such perceived advantages, the take up of e-exams in Norwegian higher education is so far limited, though varying from institution to institution. In our own university, the situation is as follows:

- For home-exams it may be common to have students use their own PCs, but then without any e-exam tools, rather just writing their answers in a word processor and submitting through the Learning Management System. Home-exams however have little defense against cheating except plagiarism control of the submitted answer.
- Proctored school exams are still paper-based. Students with special needs (e.g., hand injuries or strong dyslexia) are sometimes given the benefit of using a university-provided PC, writing their exam answer in Word rather than by hand, but in the end printing the answer and submitting on paper.

NTNU has run some limited pilot tests using digital exams in courses up to 50 students, trying out e-assessment software from two different vendors. The Rector of NTNU has however presented an ambitious goal that all exams should be digital by 2019, assuming benefits both in economy and quality, though a report suggests it may be hard to achieve full digitization that quickly [10]. In the Autumn of 2015 a normal run of digital exams will be done, though for a limited number of courses, with a limited number of students, using the tool Inspera Assessment[1]. The research questions posed in in this article are, however, not specific to that tool, but look at e-exams more generally:

- RQ1: What process improvements could be achieved with e-exams?
- RQ2: What didactic improvements in exam questions can be achieved with e-exams?
- RQ3: What requirements must be satisfied by the e-exam system to facilitate these improvements?

The rest of the paper is structured as follows: Section 2 discusses potential process improvements from digital exams. Section 3 presents potential didactic improvements. Section 4 summarizes the requirements on e-exam systems implied by the two former sections, whereupon section 5 makes a brief concluding discussion.

2. Process improvements

There are three main goals for an examination process, as suggested in Figure 1. One of them, of course, is *low cost*. A university has limited resources, and the more spent on examinations, the less left for other tasks such as preparing and improving the teaching, doing research, innovation, and dissemination. However, aiming too much for low cost can hurt other quality goals. Whenever students are going to be graded, it is seen as important that grades are *reliable*, meaning that the same performance gets the same grade regardless of the identity of the student, the sequence and time of grading, the mood and strictness of the censor, etc. One also wants the exam to be *valid* with respect to the learning goals of the course. Ideally, the exam should test all the learning goals (i.e., cover the entire content of the course), and nothing but the learning goals. Validity will to a large extent depend on the very nature of the questions asked in the exam (which is discussed in section 3) rather than on the surrounding process, but some process issues may also impact validity.

The motivation for displaying these three goals in a triangle as in Figure 1 is that they are often mutually contradictory. For instance, the preference for low cost will

[1] https://www.inspera.no/

typically pull in the direction of having few exams (e.g., one per course) of short duration (e.g., 3 hours), although this will necessarily reduce validity. A 12 hour exam would be able to cover much more of the course curriculum and with more complex types of exam tasks, but at highly increased cost for venue rental, proctor salaries, item development and grading. Also, low cost will pull in the direction of using few censors per candidate (sometimes only one, at most two) although reliability could be higher with many independent censors for each candidate (e.g., five, deleting the highest and lowest scores and averaging the three middle ones as in style judgment for ski jumping). Often, the situation might be that you can achieve two out of three goals (e.g., low cost and high reliability, but then with reduced validity), but not all three, much similar to the cost-time-quality triangle of project management.

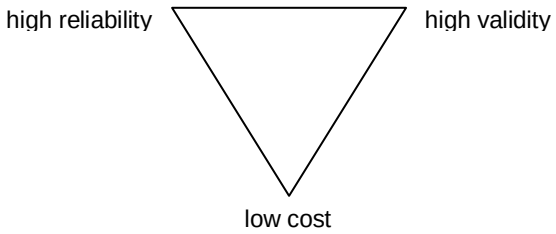

Figure 1 The impossibility triangle of exam goals

A simplified view of the current paper-based exam process is given in Figure 2. We have as start activity the teacher(s) making the question set, although there would be relevant activities taking place well before this (e.g., developing course learning goals, deciding on the type of exam, the date of exam, students enrolling in the course, doing compulsory exercises to have the right to sit the exam, the exam office deciding which rooms to use, hiring proctors, etc.). Similarly, the final activity of this diagram is the reporting of the grades, although there would be relevant activities also after this, like students seeking explanations for their grades and possibly complaining, leading to re-grading by new censors. Moreover, the teacher(s) would be likely to use exam results as one information component for their quality assurance report about the course. The motivation for dropping these various activities from the diagram is that it would otherwise be way too complex, and the version shown in Figure 2 suffices to illustrate some of the main weaknesses of the current paper-based process. This indicates several possible advantages for going digital:

Saving material costs: a considerable amount of paper is spent both for question sets and student answer sheets, as well as toner and hardware resources for printing / copying. These costs could be saved by digital exams, but there would be other technical costs instead. For e-exams to scale to large classes and peak exam days, a BYOD approach (Bring Your Own Device, i.e., each student has to bring a portable PC to the exam room) is probably the only feasible one [11], though with some increased security risks compared to having the students use institutional computers [12-13].

Process simplifications: the usage of paper demands a lot of activities that would otherwise not be needed. In Figure 2, all the activity nodes with dashed borders (9 of 17 nodes) have to be there solely because of the paper-based process. Avoiding this work would reduce the number of man-hours needed to conduct exams. Also, some other activities that are not dashed (i.e., still needed with a digital solution) would be simplified if looking more deeply into subtasks. One example is "Report grades" where teachers currently have to fill in and sign paper-based grade lists although they often have the grades on file, for instance in a spreadsheet, after scoring the answers of all

candidates. Subsequently, the administration receives the grade lists and will type grades into the system. This consumes unnecessary man-hours, and also increases the possibilities for errors (e.g., teachers writing the wrong letters for some students when copying grades from spreadsheet to paper, or administrators typing the wrong grade afterwards), so a solution where the teachers could report grades directly into the administrative system could both save time and improve quality.

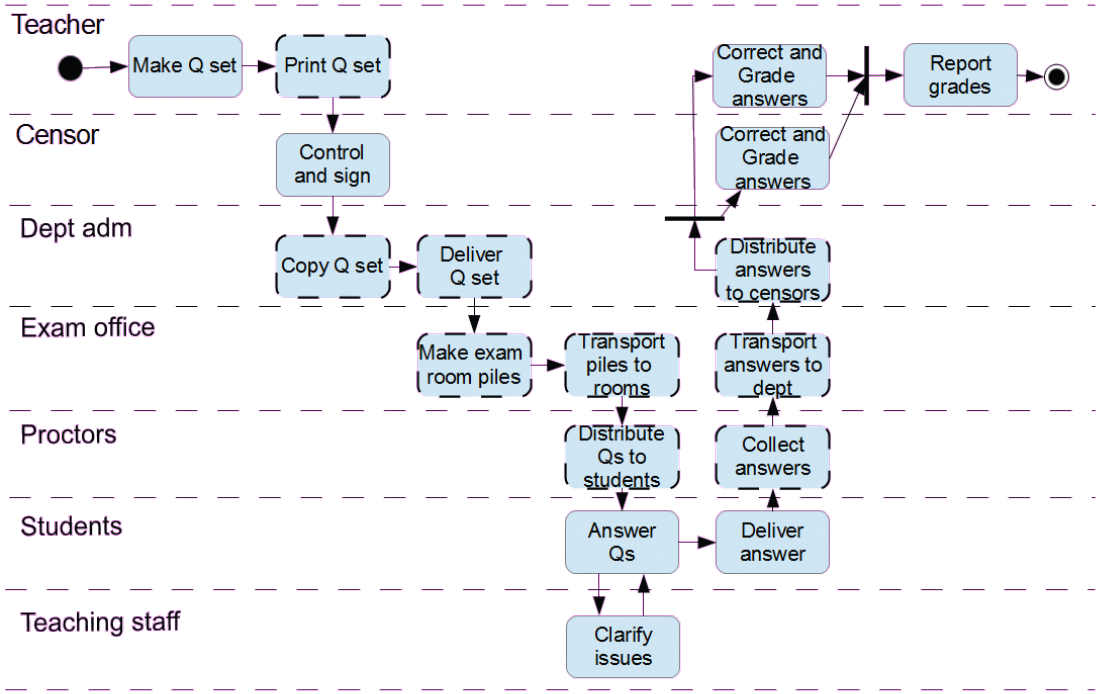

Figure 2 Traditional paper-based exam process

Better avoidance of errors in questions and question set delivery: Unfortunately, we have no data for the frequency of various errors with exams, but the following are types of errors known to have occurred:

- Teachers making errors in problem formulation, in the worst case such that parts of the exam is impossible to solve for the students.
- Discrepancy between language versions. In many courses, questions must be available in 3 languages (Bokmål, Nynorsk, English), and in some cases it has accidentally happened that one language version e.g. lacked a hint or had a problem formulated in a more difficult way than another version.
- Copying errors. In a large copying job it can go unnoticed that some copies in the batch were of poor quality, e.g. lacking a page or having weak print. Rare worst case scenarios are massive copying errors affecting all candidates of a course, such as a double-sided original run through single-sided copying so half the pages were missing, or the solution accidentally copied together with the questions and distributed to the students, rendering the exam meaningless.
- Distribution errors. Candidates for one course may be spread across several exam rooms, and one exam room may seat candidates for several courses, which necessitates the "Make exam room piles" activity by the Exam Office (i.e., for each room to get the adequate number of question sets for the right courses, and in the right language versions). Failures here or in transport or distribution may leave some students without their required question set at the start of the exam.

Going digital would eliminate copying and distribution errors, since these activities were related to paper and would no longer be needed. The two former problems are supposed to be avoided by the "Control and sign" task in Figure 2, as it is required that a person who was not involved in writing the exam questions shall look through a printed question set and sign it on the front page before it is sent to copying. Sometimes this procedure catches mistakes, sometimes not. There could be various reasons for this. Sometimes the exam set may be looked through too quickly, as a result of time pressure. In the worst case, it is less than an hour left until the exam set must be sent to copying, the teacher(s) responsible for making the exam just need a quick signature to meet the deadline, and the controller provides this signature more on the basis of trust than a thorough inspection of the question set. As for language discrepancies, it could be that one version was looked through thoroughly, but the others then just quickly. If a digital exam tool containing the question sets has explicit support for the "Control and sign" task, this could help quality in at least the following ways: (i) different language versions could be displayed side by side, to help discover any discrepancies. Possibly, automatic content analysis could be used to highlight suspected discrepancies. (ii) the controller could be guided or forced to look at each sub-problem and check that it is ok, rather than just putting a signature on the front page of the whole exam set. (iii) the e-exam tool could have a deadline for sending the exam set(s) to the controller (e.g., 3 days before the copying deadline), and either remind or enforce question writers to meet this deadline, so that the controller has sufficient time to do a proper job.

Quicker and fairer error correction during the exam. In spite of possible improvements above, it must be assumed that also e-exams will sometimes be launched with serious problems in question formulation that escaped quality control. Such errors must if possible be corrected during the exam, cf. the "Clarify issues" task of Figure 2. With paper-based exams it often means that the teacher must hurry to the exam venue to deliver the correction to the students in written or oral form, though with a written correction it might also be possible to email it to exam office personnel at the exam venue to print, copy and distribute to candidates. It can easily take an hour from the error is first discovered until a correction is delivered to all students. Also, there may be issues of fairness, especially if the teacher has to deliver the correction face-to-face. Candidates may be spread across many rooms, meaning that some will get the information sooner, others later. Assuming an e-exam tool supports teacher changes to questions during the exam and/or teacher broadcast messages to students, corrections could be made quicker and to all students at the same time, improving both validity and fairness of the exam, plus reducing stress for the teacher who no longer needs to rush to the exam venue in case of such errors.

In most exams, there are no serious errors in the question set as discussed in the previous paragraph. Yet, teaching staff is required to be available during the exam to respond to questions from the students. For a small class, it may suffice that one teacher makes one or two brief visits to the exam room(s). If candidates are spread on several rooms, and especially if these rooms are far apart, these question rounds will still take some time. For large classes, one or more teaching staff members will be more or less continuously occupied for the duration of the exam, due to the amount of questions. For instance in the Introductory IT course at the NTNU, with 1800 candidates, we might have 6 persons available at the exam venue during the whole exam. Requests for clarification may be raised because the student believes there is an error (though often there is not), the problem formulation may be a little ambiguous, or the student is simply fishing for a hint. In the latter case a "no comment" response will be likely, while in other cases the teacher might paraphrase the problem or clarify its intended

scope. The traditional face-to-face approach has several risks. Some students may get hints on issues where others get "no comment". In large classes this could be because different teaching staff members have different thresholds concerning what is a "no comment" question. Even with a single teacher situation, there could be differences in student charm or "asking skills" causing some to get a hint that others do not get. Finally, some will get information because they asked, while those who did not ask do not get it. This might seem fair, but there can be several reasons why some students do not ask even if they perceive the exam questions as unclear, e.g., shyness, fear of appearing stupid, or a belief that the teacher would deem it a "no comment" question. Again, going digital could allow broadcast clarifications. Even if just one student asked, the response can be made available to everybody if it is considered relevant and useful for everybody. This would likely reduce the number of question instances a lot, as by experience, many candidates have the same or similar questions during clarification rounds, and with broadcast responses, only the first student needs explicitly ask. Also removing the time spent just walking from room to room and from desk to desk, the online solution could make a single teacher able to handle exam clarification for much larger classes than today.

More flexible parallel grading. Electronic exam answers can be sent to several independent censors at once. For paper exams this either requires scanning or copying, or having the students write their answers on multi-sheet carbon-copy paper, increasing labor costs and/or paper costs, plus often giving poor readability of the copy, especially if students write with pencils.

Enabling problem-oriented marking of the answers, rather than candidate-oriented, both in terms of *sequence* and *work-division*. A candidate-oriented scoring sequence would mean that a censor who is supposed to score (or grade) a pile of exam answers will do this candidate by candidate, e.g., first scoring all the answers of candidate #1, then all answers of candidate #2, etc. Choosing a problem-oriented sequence, the censor would instead start by scoring all answers to Problem 1, then all answers to Problem 2, etc. With a paper-based exam, where answer sheets are naturally sorted by candidate, not by problem, the problem-oriented approach is hard to achieve because it would require the manual splitting and re-merging of each candidate's answers. For a digital exam, however, it should be easy to offer censors a simple choice between a candidate- and problem-oriented sequence, given that the e-exam tool receives the answer for each sub-problem as a separate input field rather than receiving the entire exam answer as a single file. As long as problems are formulated such that they can be evaluated independently (i.e., what the student should answer for Problem 2 does not depend on what he/she answered for Problem 1), the problem-oriented approach is believed to have several advantages:

- quicker scoring without loss of reliability, due to fewer context shifts for the censor (e.g., can concentrate just on Problem 1 and look at answers to this from 100 students, then turn to Problem 2, …)
- possibly also higher reliability due to more consistent scoring of the problem across all students, not the least due to elimination of within-exam halo effects. The halo-effect [14] is a reliability threat to grading, meaning that a student who has made a good first impression may have an unfair advantage in subsequent grading compared to students who made a poorer first impression. The halo-effect can occur between tests (e.g., a student who did well on the first test, gets unfair advantage on later tests) or within tests (e.g., a student who did well on Problem 1 gets unfair advantage on subsequent problems). Anonymous candidate numbers can protect against between-test halo-effects (unless the

same candidate numbers are used for all tests), but give no protection against within-test halo-effects. An e-exam tool set up for problem-oriented scoring with scrambled candidate order could however offer such protection, because the censor does not have to see or know what the candidate has done on previous problems, but can concentrate just on the answer to be scored in isolation from everything else.

The case for problem-oriented rather than candidate-oriented scoring is even stronger when considered as a work-division strategy, for which the difference between the two is illustrated in Figure 3. Assume that the class is too large for one censor to grade everything, for instance 300 students while each censor (or censor pair) is only expected to grade 100. Again, problem-based work division should be easy to set up for an e-exam, while it would be much harder with paper. In addition to the advantages already mentioned above, the problem-based work division will also reduce challenges related to variation in strictness between censors. Assume for instance that censor (or censor pair) X is kinder, Y is average, and Z is stricter, so candidates #1-100 may be somewhat lucky with their grades, and #201-300 similarly unlucky. The process could try to address this, suggested by the "Adjust scores?" step, but this may be tricky. It can sometimes be hard to know whether Z was really stricter or incidentally got poorer answers to grade, and attempts to adjust scores could also lead to overcompensation. In many cases, time pressure (3 week deadline for grading) will also inhibit any adjustment attempts, so scores tend to be used unaltered even if there may be a feeling that some students were a little lucky with kind censors and others were unlucky. With the problem-oriented work division, on the other hand, all candidates are scored by the kind, average, *and* strict censors. Another advantage is with advanced courses where the censors may be strong in different topics related to the course, the problem-oriented division then allows for each censor to deal with problems where the censor has expert knowledge, rather than everybody looking at everything.

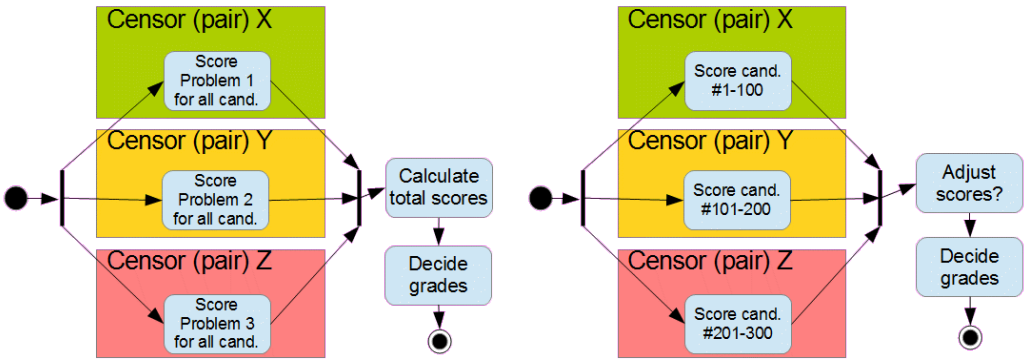

Figure 3 Problem-oriented (left) and candidate-oriented (right) work division in grading

2.5 Automated grading

For most university exams, it will not be realistic to achieve fully automated grading. The exception would be exams which are entirely composed of multiple choice or similar questions with a limited response space, and where some answers are definitely correct and other answers are definitely wrong. Here, an e-exam tool would have the obvious advantage of calculating scores automatically where a paper-exam would have to be scored by manual counting. However, even for exams where human analysis is

needed to assign grades or scores to the answers, there is a lot of semi-automated support that can be possible:

- *For short-answer problems:* possibility to score by *different answers* rather than by different candidates. Assume a programming exam contains a question like "What will be the output when running the following code? (…code)" with the correct answer being the list [1,4,9,16]. Whether problem-oriented or candidate-oriented grading is used (cf. previous subsection) the traditional approach would be that the censor looks at the answer of each student and assigns a score, i.e. if there are 100 students taking the course, the censor would assign 100 scores, though many of the answers would be similar. A much quicker way to score such problems would then be if the e-exam tool gives an overview, e.g., reporting that 35 students answered [1,4,9,16]; 18 students answered 1,4,9,16 ; 15 students answered [1,4,9]; 12 students answered blank; 10 students answered [1,2,3,4], … 1 student answered compilation error. The censor could then decide quickly that all who gave the perfectly correct answer get full score, and all who answered blank or completely incorrect (e.g., compilation error) would get zero score. Besides, decisions on how much to award intermediate answers could be made once and for all for each alternative. This would both save time and increase consistency. For instance, a decision to deduct only one point for lacking the brackets but two points for lacking the last number of the list would then be enforced for all candidates at once, whereas with a traditional scoring process, the censor's strictness might drift during the process due to changing expectations.
- *For longer questions, possibility to define typical positive or negative elements* found in an answer, and assign scores or deductions related to these. For programming exams there may typically be two ways of scoring solutions. For solutions of a reasonable quality, one might start with a full score for a problem and then deduct points for various errors found (e.g., failure to initialize a variable, failure to close a file, loop doing one iteration too much, …). For solutions of poor quality, it might be easier to start from zero points but then give some points for fragments of code that at least did some of the job (e.g., at least making the appropriate function heading and opening the file correctly, although the candidate was not able to read anything from the file). With an e-exam tool, the censor could define such codes with associated scores (plus or minus) and assign them to key combinations to annotate program answers and get proposed scores from the application. Using pattern matching and machine learning, the e-exam system might even be able to suggest annotations automatically for the censor to review (though accuracy would have to be quite high for this to be more efficient than the censor making the annotations). An additional advantage of this might be that it would ease the subsequent delivery of feedback to students demanding an explanation for their grade, as this could in many cases be achieved automatically from the annotated scores / deductions.
- *Possibility to test-run* answers to programming tasks. If developing a test suite of input data for each programming question, student answers could be run to see if they work correctly. In many cases, this could not be used for grading or scoring directly, as there may be cases where the program works for the entire test suite but still does not deserve full score. In an algorithm course, an inefficient solution would normally get a low score even if it gives the right answer, and in OO programming, failure to have an OO structure would cause deductions. Also, there could be cases where a program does not work for any

test, but was fairly close to working, thus deserving a reasonably good score in spite of its failures. Nevertheless, results of the test runs could be used to cluster student answers, e.g., answers that passed all tests in one group, those that passed tests 1,2,3 but not 4 in another group, etc. Properties such as running time, code length or structure could also be included if these are important in the course. Automatic testing and other analysis of the code will help increase reliability of the grading process in several ways. It will help the censor notice errors in the code that could otherwise be missed. It can prevent censors from mistakenly believing there is errors in code (e.g., because it differs from the proposed solution) when the code actually works. It will also speed up the grading work since it will relieve the censors of tedious manual reading of code to find out whether it would work or not.

Indeed, automated assessment of programming assignments has been a research topic for quite some time [15], so going into details about various approaches for this would be way too lengthy for the current paper. It has also been used at the NTNU, but in the context of exercises [16] rather than exams.

3. Didactic improvements

One frequent criticism against traditional pen and paper exams is that handwriting is not common in most jobs anymore. For programming this argument is especially appropriate, not just for the issue of typing versus handwriting, but for the tool support that modern programmers would normally have available in their job. The students are exposed to programming environments in the exercise part of a course, helping with detection of syntax errors, formatting, compilation, running, and debugging. With paper-based exams such tool support is suddenly unavailable, making the situation appear constructed and unnatural, thus giving an exam with reduced validity. Of course censors know that the absence of tools makes it difficult for students to avoid syntax errors and will thus not punish them too much if a piece of code appears mostly well thought out except for some small slips. But it can be hard to know for a censor what is just a slip and what is a more fundamental lack of skill. For an e-exam where the student is able to run the code and see the result, it can more confidently be concluded that failure to produce running code is not just accidental but due to lack of skill.

The ability to use industry relevant tools is an improvement in itself, but also leads to another maybe even more important improvement. Exams nowadays typically last for only 3-4 hours. This means that the exam situation will focus on small problems, whereas in real life the problems tend to be bigger. While restricting questions to small problems could be okay for an introductory programming course, it could be considered unfortunate if the students continue to be assessed mainly on problems of a quite limited size all the way through their studies. An essential challenge of the 3-4 hour paper-based exam is that it requires

- *short problems*, limited to what the students can read through in, say, half an hour (since most of the exam time should be used for answering, rather than reading the questions)
- *short answers*, limited to what the students can hand-write in 3 hours. Of course, some students can produce essays of 30-40 handwritten pages in a 3 hour exam and consider these answers long ones compared to those of other candidates, or other types of exam questions. The point here is however that these answers are still short compared to work-life reports or similar for which such exams might intend to train the students.

Even if students do not read text more quickly on screen than on paper, and do not type faster than they write, e-exams will make it possible to give longer and more complex problems and expect longer / more complex answers in courses where this is relevant. Longer problems can be given because the information made available to the student no longer has to be read sequentially but can instead be searched automatically, perhaps also analyzed in various ways depending on the type of content. It should be noted that "longer problems" does not necessarily mean that the questions themselves are longer - these are preferably concise so that it is clear to the students what they are supposed to do during the exam. However, problems can still be longer in terms of presenting bigger attachments / cases that these questions relate to. Longer and more complex answers can be expected because some content could be generated automatically or reused from digitally available material. In programming exams, traditional question types are typically

- *program comprehension tasks*, e.g. asking what will be the output if running a given program with some given input, or more generally what a program does.
- *program correction tasks*, where a given program is supposed to fulfill certain requirements but does not work due to some errors, and where it is the student's job to find and correct these errors.
- *program writing tasks*, where it is the student's job to write some piece of code, often from scratch - although sometimes such tasks say that the student can assume the existence of some useful function or class that can do part of the job.
- *program design tasks*, where the students are not expected to write the full code for something, just a code skeleton (e.g., class definitions, method definitions) to indicate their preferred design.

Program comprehension tasks in their basic form would be meaningless if the student is able to run the code (which would directly give away the answer), so in an e-exam such questions would imply that the usage of a programming environment must be blocked until these are answered.

Program correction tasks, if at all given, tend to involve very short pieces of code (e.g., 10-20 lines) because longer programs would be hard to read through in a short time and cause too high copying costs. In work-life, however, one would normally have to look at considerably bigger chunks of code when doing debugging work. With e-exams, long program attachments would not have challenges with paper cost. Moreover, if the students are able to use tools for debugging, static and dynamic analysis and to compile and run their proposed corrections, they could be expected to deal with larger chunks of code than for the paper-based exam. This would increase the exam's resemblance of professional work and thus increase the validity.

Similarly for program writing tasks, these are normally made to result in fairly short programs, and often the division in sub-problems reduces the amount of code that has to be written at each sub-problem. Alternatively, program design tasks could explore the ability to structure somewhat larger problems (that might have resulted in several hundred lines of code if implemented fully), but then in a shallow way, exploring only the code skeleton, not the full running code. To be really proficient in programming a candidate would however be expected to be able to write longer chunks of code, i.e. requiring the combination of program design and coding. With an e-exam this might be possible, e.g. making questions demanding more code, but at the same time making some code available in a library (without leading the students too much, i.e., some of the code in the library could be relevant, some not, and it might not be said exactly what should be used and where). This would make programming tasks more realistic and more representative of the skills that future employers would be looking for.

4. Discussion and conclusion

Our first research question was about the potential process improvements of e-exams As outlined in section 2, in addition to the obvious improvements (saving costs by eliminating paper and paper-related activities, and satisfying students preferences for typing rather than handwriting), e-exams can provide a lot of additional process improvements that could also ease the work for teachers, both in preparing for the exams (e.g., making and quality controlling questions), conducting the exams (e.g., providing clarifications online rather than by face-to-face rounds in the exam rooms), and grading (both by offering problem-oriented scoring as a real alternative to candidate-oriented scoring, and by automated support for scoring). Our second research question was about how digital exams can improve validity of the tests. Although any examination situation will likely be somewhat artificial compared to the real world candidates are being prepared for, it was suggested that the students' work process during the test can be made more similar to real-life work. For instance in programming and other related informatics courses, the usage of tools fitting the task can make the students' work with exam questions more realistic. Questions themselves can also be made more realistic because the digital form can allow longer and more complex questions and answers, and students could be able to test during an exam whether e.g. their programming answers actually solve the problem.

While the requirements provided in previous works such as Sclater and Howie [2] are relevant, the requirements lack a little detail on some important aspects. Specific system features pointed out in this paper are:

- Support for checking correspondence between different language versions of the exam questions.
- The possibility of online clarifications rather than doing this face-to-face. This could be relevant both for serious errors in a question set (so that a correction is necessary) and for questions that the students perceive as vague (where a clarification might not be necessary, but would be nice to have, and if given, should be to equal benefit of all candidates).
- The importance of offering a choice between problem-oriented and candidate-oriented grading.
- The possibility for semi-automatic grading based on clustering of answer content and automated analysis of the answers. In informatics, the running of programs through test suites would be one example.
- The possibility to integrate relevant work tools with the e-exam tool. In the informatics field, typical examples would be programming environments, modeling tools, and project management tools.

With a high level of ambition, it is not likely – and maybe not the best – to expect that a single e-exam application should be able to satisfy all requirements one might have. Satisfying all needs will be almost impossible because there will be different requirements in different universities / different countries (e.g., different regulations for exams, different grading systems, different rules for grade complaints, etc.), different needs for different subjects or courses (e.g. various types of tools that one might want the students to be able to use), different question types, for different types of students, different pedagogical styles, etc. A better approach seems to be to have several interoperable tools satisfying the needs together, which will give better flexibility and more choice of alternative services, a kind of e-exam or e-learning ecosystem [17]. More than satisfying all kinds of requirements it is therefore important that the e-exam applications chosen by Norwegian universities have open interfaces with well-

documented API's, so that it is easy to make add-on applications to provide extra services. Ideally, an e-exam solution would already have an active ecosystem around it, both concerning add-on services and sharing of question ideas.

References

1. Brusilovsky, P. and P. Miller, *Web-Based Testing for Distance Education*, in *World Conference on the WWW and Internet (WebNet '99)*. 1999, ERIC: Honolulu, Hawaii.
2. Sclater, N. and K. Howie, *User requirements of the "ultimate" online assessment engine.* Computers & Education, 2003. **40**(3): p. 285-306.
3. Kuikka, M., M. Kitola, and M.-J. Laakso, *Challenges when introducing electronic exam.* Research in Learning Technology, 2014. **22**.
4. Conole, G. and B. Warburton, *A review of computer-assisted assessment.* Research in Learning Technology, 2005. **13**(1).
5. Sim, G., P. Holifield, and M. Brown, *Implementation of computer assisted assessment: lessons from the literature.* Research in Learning Technology, 2004. **12**(3).
6. Csapó, B., et al., *Technological issues for computer-based assessment*, in *Assessment and teaching of 21st century skills*. 2012, Springer. p. 143-230.
7. Gipps, C.V., *What is the role for ICT-based assessment in universities?* Studies in Higher Education, 2005. **30**(2): p. 171-180.
8. Kuo, C.-Y. and H.-K. Wu, *Toward an integrated model for designing assessment systems: An analysis of the current status of computer-based assessments in science.* Computers & Education, 2013. **68**: p. 388-403.
9. Terzis, V. and A.A. Economides, *The acceptance and use of computer based assessment.* Computers & Education, 2011. **56**(4): p. 1032-1044.
10. Hovde, P. and S.O. Olsen, *Utredning - Digital eksamen NTNU 2015-2019*. 2015, NTNU: Trondheim, Norway.
11. Hillier, M. and A. Fluck, *Arguing again for e-exams in high stakes examinations.* Electric dreams. proceedings ascilite, 2013: p. 385-396.
12. Dawson, P., *Five ways to hack and cheat with bring-your-own-device electronic examinations.* British Journal of Educational Technology, 2015.
13. Sindre, G. and A. Vegendla, *E-exams versus paper-based exams: A comparative analysis of security threats*, in *Norwegian Information Security Conference (NISK 2015)*. 2015, Bibsys OJS: Ålesund.
14. Malouff, J.M., A.J. Emmerton, and N.S. Schutte, *The risk of a halo bias as a reason to keep students anonymous during grading.* Teaching of Psychology, 2013. **40**: p. 274-280.
15. Ala-Mutka, K.M., *A survey of automated assessment approaches for programming assignments.* Computer science education, 2005. **15**(2): p. 83-102.
16. Trætteberg, H. and T. Aalberg. *Authoring specification-and test-based Java exercises with JExercise*. in *Norsk Informatikkonferanse (NIK 2007)*. 2007. Trondheim: Tapir.
17. Uden, L., I.T. Wangsa, and E. Damiani. *The future of E-learning: E-learning ecosystem*. in *Digital EcoSystems and Technologies Conference, 2007. DEST'07. Inaugural IEEE-IES*. 2007: IEEE.