A System for Home Exercises Addressing the SOLO Taxonomy, Fairness and Deep Learning

Martin Hell and Paul Stankovski

Department of electrical and information technology,
Lund University, Box 22100, Lund, Sweden
{*martin.hell, paul.stankovski*}*@eit.lth.se*

Abstract

This paper describes the assessment of students' learning in the course "Advanced Web Security"at Lund University. The goal was to create an examination process that was individual for each student, fair, and inspired by the different levels in the SOLO taxonomy. The assessment was based on home exercises and the rationale behind the design of the problems as well as the evaluation results are presented.

1 Introduction and background

The Advanced-level course "Advanced Web Security" is a 4.0 credit non-mandatory course offered primarily to students on the C and D programs at the faculty of engineering, Lund University. The course was first given in the fall of 2012 and as it was to be developed from scratch there was much freedom in choosing the type of assessment for the course. Most courses at LTH are assessed by means of a written exam by the end of the course, sometimes in combinations with projects and/or laboratories. While written exams are comparatively cost efficient, they have several drawbacks.

- The time limit on a written exam may cause student to perform worse than their actual capacity.
- Illness or other factors may affect the result for some students.
- Students study to pass the exam, affecting their possibilities for deep learning.

- Only a fraction of the topics discussed during the course can be tested on the exam due to the relatively short time frame in which it must be conducted.

On the other and, it is fair and it is easy to guarantee that solutions were worked out by the student alone. It was decided to do the majority of the assessment through hand-in exercises instead, hoping to provide the student with a way of avoiding the above issues. There was also one mandatory lab which will not be discussed here.

The SOLO toxonomy is one way of structuring the levels of understanding that a student go through when learning a subject. In SOLO there are five levels.

1. Prestructural - The student misses the point.
2. Unistructural - The student has picked up on terminology and can give simple explanations and connections.
3. Multistructural - Several aspects of a topic can be explained.
4. Relational - The student can compare the different aspects of a topic, relate them to each other and apply them. The student can also see how the different parts together make up a whole.
5. Extended abstract - The student can use the relations between the different aspects of the topic and extend it by generalizing the underlying ideas and transfer the principles to new situations.

The different levels of the taxonomy were used as a basis for the structure of the hand-in exercises. The course material consisted of lecture notes and lecture slides.

2 The examination form

Assessment can be formative or summative. A formative assessment takes place during the course and can be more integrated with the learning as students can use the result of the assessment of one part when learning the next. On the other hand, it is more difficult to integrate several parts of the course. This is easier in a summative assessment, which instead has the drawback that students tend to focus their studying to the end of the course.

The type of the assessment is very important for the learning. When, how and what the students study will depend on the form of assessment. Hand-in exercises, similar to a home exam, prevents the students from just memorizing facts and they can instead focus on understanding the material, facilitating deep learning. Moreover, the time pressure is much less and the students can to some extent decide when to solve the exam problems. Hand-in exercises during the course will also make the assessment a part of the learning process. The main drawback is the risk of plagiarism. It is much more difficult for the teacher to verify that it is actually the student that signed the exam that also solved the problems. Another drawback is that students may focus on only learning what is asked for and misses several important parts of the course material. Some of the hand-in exercises that will be used has the form of small projects. A project has the advantage of allowing the problems to be set in a more practical setting. This increases the students' motivation, again facilitating deep learning.

3 Creating home exercises

The home exercises were divided into three distinct types, each with their own assessment criteria.

- **A-type:** These problems were rather straight-forward and aimed at checking that the student had read and understood the material.
- **B-type:** These problems required the students to look for information outside the given material.
- **C-type:** These problems required more work that the other types. It could either be to write a program to simulate something, but it could also be to write a short technical report that considered one part of a lecture put into a wider context.

The A-type problems could usually be answered with one or two sentences. They were primarily focused on assessing both declarative and functional knowledge in the quantitative phase of SOLO, i.e., the unistructural and multistructural levels of understanding. Some problems were also assessing declarative knowledge in the relational level of SOLO, e.g., comparing and contrasting different technologies, and explaining certain aspects of some topic.

The B-type problems typically required a longer solution, sometimes writing a small program and sometimes looking at a related topic and compare it to what has been discussed on the lecture. These were intended

to have a clear focus on allowing assessment of declarative and functional knowledge in the relational level of SOLO, i.e., a part of the qualitative phase.

The C-type problems were in the spirit of a mini project. The students had to understand a certain topic at a level that allowed them to either write a computer program to simulate a certain situation, apply a technology in a practical situation, generalize it to include new ideas or reflect on certain aspects of a specific technology.

Applying the SOLO taxonomy was only one aspect in designing and classifying the problems. Attention was also given to the law of higher education (Högskolelagen) which specifies that a course on advanced level should (among other things) develop the student's ability to work in an environment with high demands on independency. The idea behind the B- and C-type problems was also to a large extent based on this.

3.1 Solving home exercises

The most straightforward way for assessment is to let the student solve all problems. This has several drawbacks that we wished to avoid.

- Friends that usually work together will solve the problems together. We wanted to at least make it probable that they have worked out solutions individually.
- Every student have different interests. We wanted to, at least to some extent, let them choose which problems to work on. If they work on problems they have chosen themselves, there is a higher probability that they have an interest in what they are doing. Then they will put more effort into the solution and also learn more.

There is a tradeoff between these two aspects. If we let them choose which problems to solve, then friends will choose the same problems and cooperate. Another aspect is the advantage of solving problems in small groups, allowing the to discuss and reach a solution together. This was solved by making a separation using the different types of problems. A-type problems where individually solved and could not be freely chosen, while C-type problems could be chosen by the students. They were also allowed to work in pairs on the B- and C-problems as these were more difficult and required more time.

3.2 Problem selection

Several different variants of grading were discussed before the course started, but we settled for a scoring system that was very easy to understand despite different classes of problems and differences between the classes.

One problem set consisted of 8 A-type, 4 B-type and 1 C-type problem. The selection of problems were done in the following way.

- The 8 A-type problems were randomly selected from a set of 16 problems. These were solved individually and determined using a SHA-1 hash of the student's personal number together with fixed string (hash salt) that was changed for each problem set. Thus, if two students had several overlapping problems one week, the overlap was changed the next week. This provided good randomization, both for problems and for overlaps between students, but also added a nice extra feature; we did not have care about that if problem i in one set seemed a bit difficult we should make problem i less difficult in the next set in order to stay fair. This was automatically solved by probability theory. The use of a standardized hash function allowed anyone to easily reproduce th choice of eight problems, for any student, any week, using a one-line Linux command.

- The B-type problems were fixed and solved by everyone in groups of two. Thus, all groups solved the same problems. As these required more work than A-type problems, it was seen feature rather than a problem that students could discuss these not only within the group, but also between groups.

- The C-type problem was chosen by the students from a set of two problems. Typically, one problem was a programming exercise while the other problem consisted of studying some topic in more detail, putting it into a new context or comparing it with similar technologies. The findings were summarized in a 2-4 pages report.

3.3 Grading

Each A-problem was awarded 0-1 points, each B-problem 0-2 points and each C-problem 0-8 points. Thus, they could get 0-8 points on each problem type. With in total 5 problem sets, the maximum number of points were 40 for each problem type. The grading was done as follows.

- **A-type:** Grade 3: 25, Grade 4: 33

- **B-type:** Grade 3: 20, Grade 4: 28
- **C-type:** Grade 3: 20, Grade 4: 28

The rationale behind this was the A-type problem tested more basic knowledge and should require higher score for a given grade compared to the other problems. Moreover, as these were solved individually, we could better argue that each student had actually worked through most of the material in the course. Grade 4 was given if all types summed up to this grade. Grade 5 could be achieved by taking an oral exam, provided the home exercises were graded 4. (For those very close to a certain grade for one type, we provided a way to complement to that grade.)

4 Evaluating the process

The course evaluation was divided into one formative and one summative part. The goal of the formative part was to evaluate the course during its execution, allowing real-time adaptations and improvements. This was done using weekly or bi-weekly informal discussions with course representatives (two students), and also by taking the opportunity to discuss the home exercises during feedback sessions 5.

The summative course evaluation was conducted using the general CEQ, but with the addition of a few course specific questions.

4.1 Formative evaluation

Results from the formative part quickly revealed that the course required much more work than anticipated, and we decided to remove one problem set from the original six.

Since there were four B-type exercises, it was common that students divided them among themselves and solved two each. This was perhaps related to the heavy work burden, but this strategy would probably be used by many students even if the workload was lighter. Interestingly, this was one of the aspects that we explicitly tried to avoid by making the A-type problems individual. Making also the B-type problems individual would of course address this problem, but it would make it much more difficult to have a fair assessment, and the teachers' workload would become much heavier.

4.2 Summative evaluation

Our conclusions from the summative part were that the home exercices were very appreciated. One course specific question was related to home exercises and was formulated as.

> *I would have preferred a written exam at the end of the course instead of hand-in exercises.*

This question received an average score of -84 (where the choices were [-100 -50 0 50 100]). The question was of course very generally stated, and one must be careful when drawing conclusions about our specific system for conducting the home exercise assessment. Still, as the question was given in the context of the course, the students seem to have appreciated the system. The free text answers on the report gave more insight. The comments related to the our particular system of home exercises are given below (translated to English).

> *Good hand-in exercises system, much freedom of choice, which is good.*
>
> *Good strategy for hand-in exercises.*
>
> *Hand-in exercises were very good. Both the fact that we had to work both individually and in pairs, and the division into A- B- and C-type questions. It was particularly good that we could choose the C-question ourselves. This made it more fun.*
>
> *...and a good system for home exercises.*
>
> *If you attended the lectures, you could solve most A-type problems almost immediately.*
>
> *The C-type problems requires us to apply the theory in a practical setting. This happens otherwise too seldom according to me.*
>
> *The programming problems where you put the theory into practice.*
>
> *The home exercises forced you to go through all course material at least once at some point and understand the majority of each topic.*

It was unfortunate that it was not possible to get grade 5 on the home exercises.

5 Feedback to students

Feedback is an important part of the assessment. The grading must be motivated and the students must have a chance to learn from their mistakes. Still feedback is very resource consuming and with many students, grading of the problems will alone consume most of the time can be put into the course, economically speaking. While feedback in general is important, its usefulness for a given student is likely to vary. Instead of giving feedback for each student, we allocated office hours every week when students could come and discuss their solutions and ask about the grading. Thus, we provided no automatic feedback except the actual score for each problem, but gave them the opportunity to receive detailed and extensive feedback if they wanted to. The result of this experiment was somewhat surprising. Only very few student took the opportunity to get feedback during office hours even though we were careful when scheduling the office hours so that it would not collide with other courses that most students took in parallel. Only about five students visited the feedback session. On the other hand the students that did visit the sessions typically did it every week.

To get more information about this behaviour, two course specific question were added to the CEQ, see Figure 1.

It seemed that the students were in general satisfied with the possibility to get feedback, though some did find it unsatisfactory. Students preferences to which feedback strategy is best is more divided. Those that were happy with no written feedback but extensive oral feedback are as many as those with the opposite preference. Similar to learning approaches, it seems that feedback preferences are very individual.

6 Lessons learned and future considerations

Scalability is probably one of the most difficult problems to solve in an examination process that should be both individual and based on hand-in exercises. The course had 40 students which turned out to be manageable,

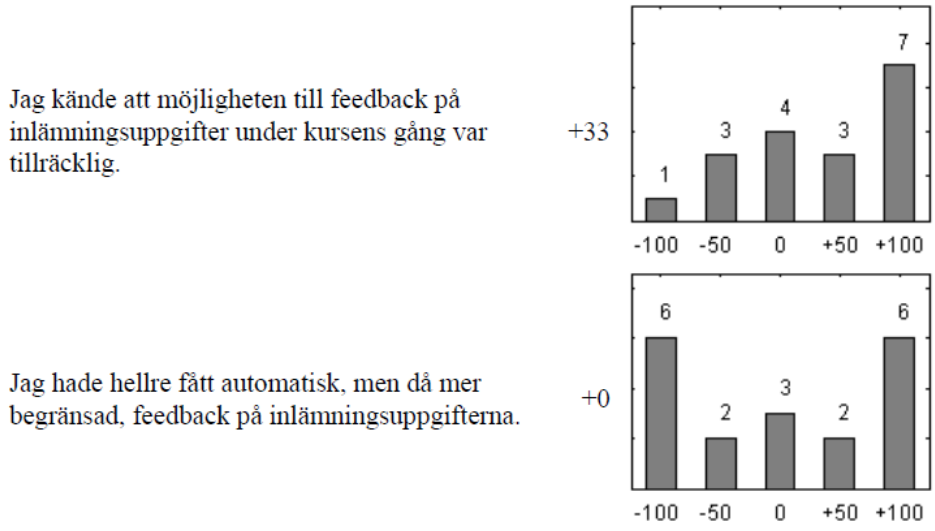

Figure 1: CEQ evaluation results for two course specific questions.

but still required slightly more time than economy allows. Computer systems like moodle can be used to automate parts of the process, but it can not be used for all things, e.g., validating program code. While it can automatically correct certain questions of multiple choice type, assessment according to the higher levels of the SOLO taxonomy can not be done using multiple choice [**?**].

We further experienced that it was sometimes difficult to categorize some problems. A problem that required a small program to be written, more involved calculations or a higher level of detail were not appropriate as A-type problems since they were more difficult as the other A-type problems. At the same time, they seemed too easy for B-type problems. Even though the different types were designed to allow a wide range of problems, our clear distinction between the A- and B-types (randomly selected and individual vs. fixed and solved in pairs) sometimes felt too restrictive when designing problems. One way to address this could be to allow students to choose from a set of B-type problems, similar to the case with the C-type.

Having a set of only 16 A-type problems and randomly choosing eight of these does give rather significant overlaps between different students (four on average). As an important feature of these problems is that they should be individually solved, these sets will be increased to 32 for the next course instance, decreasing the average number of overlaps between two students

to two instead of four.

Another interesting aspect is *when* the problems should be handed out. The A-type exercises could often be solved immediately if the students attended, and paid attention to, the lecture. If problems are handed out before the lecture, students could potentially check which A-type exercise they have to solve and then focus on just understanding the parts of the lecture that allowed to to solve "their" problems. An alternative is of course to hand them out after the lecture, but a more interesting idea could be to hand them out before, but not to reveal which eight A-type problems each student had to solve. Then they would pay attention to all parts of the lecture, while at the same time knowing exactly which aspects are considered important (assuming all important aspects are covered by *some* problem). This could very easily be achieved by just revealing the hash salt after the lecture.