Assessing Knowledge Through Written Reviews

Emil Folino Blekinge Tekniska Högskola
emil.folino@bth.se

May 5, 2017

Abstract

In this paper a method of qualitative assessment of programming student is investigated. The qualitative assessment is done by reading students' review text from individual programming projects and analyzing the content according to the SOLO taxonomy.

I. Introduction

Quantitatively assessing students in programming and computer science courses is often easy to do. Have the student completed the assigned tasks? Does the application work as it is supposed to do? Does any software tests fail? However assessing the comprehension and understanding of the completed assignments and doing a qualitative evaluation of the student is harder [1]. The SOLO Taxonomy was proposed by Biggs and Collis in 1982 and is abbreviated from Structure of the Observed Learning Outcome. The taxonomy is used to qualitatively assess students' work. The SOLO taxonomy consists of five levels of understanding: Prestructural, Unistructural, Multistructural, Relational, and Extended Abstract. In section II the five levels of understanding will be exemplified by the students' review texts.

In [2] McCracken et al. evaluated and first year Computer Science students' programming competency. A framework outlining the expectations of first year Computer Science students are proposed. However the assessment is only done in a quantitative way and there are no recommendations for any qualitative assessment method.

Paulo Blikstein [3] uses snapshots of students' code during a programming assignment together with data mining to automate a technique to assess and analyze students learning programming. It is concluded that together with other data sources (Blikstein propose interviews, tests and surveys) the automated assessment can give insights into the understanding of the learing of programming.

In [4] Lister et al. looks at written and think-aloud responses to exam questions. Lister et al. conclude that experienced programmers more frequently answered with SOLO relational responses compared to the novice programmers multi-structural responses. Lister et al. recommend that the students are given written assignments together with programming assignments making it easier to evaluate the understanding and comprehension the students obtain in programming courses.

II. Examples of SOLO levels

In this section examples of how the review texts are mapped to each level of the SOLO taxonomy are shown. The review texts have been translated from Swedish to English by the author. The original Swedish texts are found in appendix A.

i. Prestructural

No answer more than repeating the question. The student is failed based on the text. The Prestructural SOLO grade are used for students that have not handed in the projects or have failed to write a review text that explains the problem.

ii. Unistructural

The answer contains no technical description of how the solutions have been implemented.

Example: "The search feature has it's own page (accessible from the navbar) where you can search for articles and object descriptions with a word consisting of letters(a-ö) and numbers(0-9). The search results are presented in a list and the first result in an article/object description is marked with and part of the text."

iii. Multistructural

The student give a technical explanation of the implementation, more or less line by line and does not relate the implementation to other parts of the code or prior exercises in the courses.

Examples: "For every iteration in the loop an object is appended to slar.json. So when the loop is done it was just adding the last parenthesis and clean the file so it is valid JSON. I thought this requirement was kind of complicated and my solution is absolutely not the fastest but it does what it is supposed to do and that should suffice."

"I have tried to use a lot of built-in functions like 'map', 'reduce', 'filter', etc. to get the correct information from the arrays I use."

iv. Relational

The student give a technical explanation of the implementation and justifies their choices through related course material or real-world examples.

Examples: "I chose to not split my client as it is done in the Gomoku assignment [1]. I know that the reason is to separate general and domain-specific code, but I am not going to extend on this client so I have decided to put all the code in the same file."

"When i decided on the style for the page i looked at other websites with a connection to funerals."

"With earlier assignments' clients as a base I did a client that tests the server."

v. Extended Abstract

No examples of Extended Abstract texts were found in the review texts. The students are first-year students and are not asked to produce novel material.

III. Method

In our daily teaching at Webbprogramming (dbwebb.se) at BTH the students do programming assignments each week. Together with the exercises they hand in a written review, answering 3-5 questions centered around the topics and assignments of the week. At the end of each study period, a 10 week period, the students hand in an individual project together with an extended review text. The students are graded both with regards to the completed work and the review texts. The following web pages from the program's website explains how the students are graded on their review texts according to the SOLO Taxonomy: [5] and [6].

In this report the review texts of three subsequent course projects are analyzed and SOLO graded. The students are given a grade of 1-5 according to the five levels of the SOLO Taxonomy. The SOLO grading will be done manually by reading the review texts and the SOLO grading will be done anonymously, but traceable.

The SOLO grading of the review texts will be done by the author. To ensure an even level of grading the other teachers of the courses are going to do a similar grading and evaluation of a subset of the review texts.

[1]A programming assignment earlier in the course

The collection of review texts is done with a web scraper implemented in python. The web scraper fetches the review texts from the students published projects. The review texts are stored in a database together with the website url of the published project and a traceable reference to the student. The review texts are fetched in a manner that removes the names and student acronyms from the review texts to ensure anonymity in most cases.

The analysis of the review texts are done in a web form and the SOLO grade is stored in the same database table as the review texts. The web form removes all styling done by the students and due to the way the collection of review texts are done this further ensures the anonymity of the students.

After the review texts have been analyzed and SOLO graded the SOLO grade will be compared to the final grade of the course. The students' final grade will be fetched and stored in another database table together with the same traceable reference to the students. The final grade for the course and the SOLO grade can now be compared and analyzed to evaluate if there is a correlation between the SOLO grade and the final grade in the course.

As the review texts are taken from three subsequent courses the evolution of the students' understanding of the course material and programming in general can be investigated.

The web scraper and analysis web form can be found at the author's Github page [2].

IV. Results

Here results will be shown.

V. Discussion

Here the analysis and discussion of the results will be shown.

[2]https://github.com/emilfolino/pedagogy

VI. Conclusion

Here the conclusions are drawn.

VII. Future Work

Here I will discuss my plan to do it with NLP and AI.

VIII. Acknowledgements

Thanks to Mikael Roos, Kenneth Lewenhagen, and Andreas Arnesson of Webbprogrammering at BTH for helping to ensure even SOLO grading by reading and evaluating a subset of the analyzed review texts.

References

[1] Biggs JB, Collis KF. Evaluation the Quality of Learning: The SOLO Taxonomy (structure of the Observed Learning Outcome). Educational psychology. Academic Press; 1982. Available from: `https://books.google.se/books?id=ZqxSMwEACAAJ`.

[2] McCracken M, Almstrum V, Diaz D, Guzdial M, Hagan D, Kolikant YBD, et al. A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. ACM SIGCSE Bulletin. 2001;33(4):125–180.

[3] Blikstein P. Using Learning Analytics to Assess Students' Behavior in Open-ended Programming Tasks. In: Proceedings of the 1st International Conference on Learning Analytics and Knowledge. LAK '11. New York, NY, USA: ACM; 2011. p. 110–116. Available from: `http://doi.acm.org/10.1145/2090116.2090132`.

[4] Lister R, Simon B, Thompson E, Whalley JL, Prasad C. Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. ACM SIGCSE Bulletin. 2006;38(3):118–122.

[5] Roos M. Att skriva en bra redovisningstext;. Accessed: 2017-02-10. https://dbwebb.se/kunskap/att-skriva-en-bra-redovisningstext.

[6] Roos M. Varför regnar det på bergssidan - ett exempel på SOLO taxonomin;. Accessed: 2017-02-10. https://dbwebb.se/kunskap/varfor-regnar-det-pa-bergssidan-ett-exempel-pa-solo-taxonomin.

A. Original Swedish Texts

Below are the original Swedish review texts and their corresponding English translations.

Original Swedish Text: Sökfunktionen finns som egen sida(via navbar) där man kan söka i artiklar och i objektsbeskrivningar med ett ord som utgörs av bokstäver(a-ö) och siffror(0-9). Träffarna presenteras i en lista och första träffen i en artikel/objekt markeras med gult och del av texten.
Translation: The search feature has it's own page (accessible from the navbar) where you can search for articles and object descriptions with a word consisting of letters(a-ö) and numbers(0-9). The search results are presented in a list and the first result in an article/object description is marked with and part of the text.

Original Swedish Text: För varje varv i loopen appendas ett object till salar.json. Så när loopen är färdig så var det bara att lägga till de sista paranteserna och städa upp filen så att det skulle validera som JSON. Jag tyckte detta krav var ganska krångligt och min läsning är absolut inte den snabbaste men den gjorde vad den skulle och det fick vara bra nog.
Translation: For every iteration in the loop an object is appended to slar.json. So when the loop is done it was just adding the last parenthesis and clean the file so it is valid JSON. I thought this requirement was kind of complicated and my solution is absolutely not the fastest but it does what it is supposed to do and that should suffice.

Original Swedish Text: Jag har försökt att använda mycket inbyggda metoder som 'map', 'reduce', 'filter', etc. för att få ut rätt information från de arrays jag använder.
Translation: I have tried to use a lot of built-in functions like 'map', 'reduce', 'filter', etc. to get the correct information from the arrays I use.

Original Swedish Text: Valde att inte dela upp min klient som det är gjort i Gomoku. Jag vet att anledningen var för att kunna hålla isär generell och domänspecifik kod, men eftersom jag inte tänkt bygga vidare på den här klienten så lägger jag allt i samma.
Translation: I chose to not split my client as it is done in the Gomoku assignment [3]. I know that the reason is to separate general and domain-specific code, but I am not going to extend on this client so I have decided to put all the code in the same file.

Original Swedish Text: När jag bestämde stil för sidan kollade jag runt lite på andra webbplatser som har en koppling till begravningar.
Translation: When i decided on the style for the page i looked at other websites with a connection to funerals.

Original Swedish Text: Med tidigare uppgifters klienter som grund gjorde jag en klient som kan testa servern.
Translation: With earlier assignments' clients as a base I did a client that tests the server.

[3] A programming assignment earlier in the course