ALOHA - A Grading Tool for Semi-Automatic Assessment of Mass Programming Courses

Tuukka Ahoniemi
Institute of Software Systems
Tampere University of Technology
P.O.Box 553
FI-33101 Tampere, Finland
tuukka.ahoniemi@tut.fi

Tommi Reinikainen
Institute of Software Systems
Tampere University of Technology
P.O.Box 553
FI-33101 Tampere, Finland
tommi.reinikainen@tut.fi

ABSTRACT

Many mass programming courses face the problems related to using multiple graders for student assignments: achieving objectivity and consistency in grading. Grading rubrics are a possible solution to this problem. ALOHA is an online grading tool that provides rubrics that all the graders have to use. ALOHA also provides features that make the grading process, including the writing of a feedback text, more convenient for the graders and the teacher.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education

1. INTRODUCTION

Novice level programming courses in Tampere University of Technology tend to be large mass courses with hundreds of students. On one hand we would like to keep our workload of these courses maintainable but on the other hand novice students require profound and personalized feedback on their programming assignments. Fully automatic assessment would easen the workload, but does not provide the important feedback on matters that cannot be automatically assessed and have a lot of focus on novice courses, like programming style [2]. As a compromise we use semi-automatic assessment [1]. This usually requires the mass course to use multiple graders for the student assignments.

The obvious aim is to achieve a consistent and objective grading between different graders. In practice, this often turns out to be quite difficult [4]. To aid courses in this as well as in providing proper feedback for the students we present an online grading tool called ALOHA.

2. PROBLEMS OF OBJECTIVE GRADING

To ensure consistency among graders, the graders need to be properly trained for the job and they are required to use marking schemes to direct their attention to the appropriate things in a student's work [5]. Becker suggests a similar approach and also defines the schemes as *rubrics* [3].

The use of rubrics increases objectivity because the idea is in splitting the assessment into smaller parts. The grader can concentrate on a single aspect of the work instead of giving a general grade for the final work. The total grade can later be derived from the grades given to the parts.

According to Habeshaw et al. [5], the only definite way to grade objectively is by using only objective tests (multiple-choice questions etc.), but doing so would not be desirable in courses where a major part of the learning is based on the student programming by himself. By receiving in-depth feedback the student can focus on improving his weaknessnes in the future. So the only way to keep this and still achieve objectivity is to divide the assessment in small enough parts with rubrics.

Earlier we used traditional rubrics in grading but unlike Becker we did not let the students see the filled rubrics. Thus all graders did not use the rubrics properly but formed the grade before even filling the rubric. The purpose of ALOHA is to provide the rubrics online in a way that each grader has to fill them correctly and in a similar way.

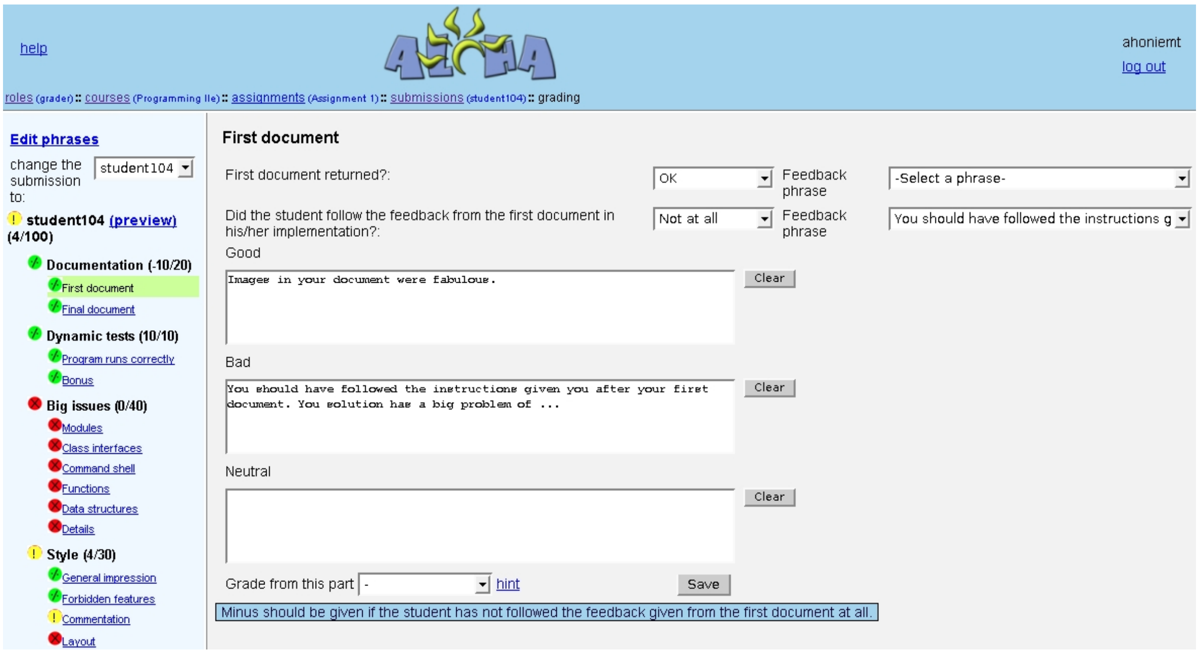
The fact that many of the dozens of student submissions are alike causes problems. The submissions resemble each other due to student guidance, but also because the assignments are kept moderately simple and very strict due to the nature of CS1-level teaching. Because of this alikeness among submissions, the grader easily starts to feel as if he is grading *the same work over and over again.* This in turn may tempt the grader from writing in-depth feedback.

3. GRADING WITH ALOHA

To help the grader in writing profound feedback ALOHA has a feature one could call *semi-automatic phrasing.* This means that when the grader gives a grade for a small subpart he can choose a ready-made phrase to be added for the student's feedback. This helps the grader not to have to write the same kind of feedbacks all over again for each of the student doing the same mistake. The phrases are related to a certain assignment and added beforehand by the lecturer who has set the rubric up. Still the grader has the ability to modify old and create new phrases.

Semi automatic phrasing has been a feature of which the graders have liked a lot and because of it they have felt that the tool is made to help them in the grading and not just ensuring that everyone is grading in a similar way. The tool is also a great aid for the lecturer to manage the whole grading process as he has all the gradings and final graders in one collective online place.

Figure 1 shows the grading view in ALOHA. This view presents the grading rubric. The grading is divided into hierarchy shown in the left consisting of categories (*Documentation, Dynamic tests, etc.*) which consist of subcategories (*First document, Final document, etc.*). Each of the subcategories is given a grade based on the *grading*



roles (grader) :: courses (Programming IIe) :: assignments (Assignment 1) :: submissions (student104) :: grading

change the submission to: student104

student104 (preview) (4/100)

- Documentation (-10/20)
  - First document
  - Final document
- Dynamic tests (10/10)
  - Program runs correctly
  - Bonus
- Big issues (0/40)
  - Modules
  - Class interfaces
  - Command shell
  - Functions
  - Data structures
  - Details
- Style (4/30)
  - General impression
  - Forbidden features
  - Commentation
  - Layout

First document

First document returned?:	OK	Feedback phrase	-Select a phrase-
Did the student follow the feedback from the first document in his/her implementation?:	Not at all	Feedback phrase	You should have followed the instructions g

Good

Images in your document were fabulous.

Bad

You should have followed the instructions given you after your first document. You solution has a big problem of ...

Neutral

Grade from this part -

Minus should be given if the student has not followed the feedback given from the first document at all.

Figure 1: Screenshot of the grading view in ALOHA

items related to it (*"First document returned?"*). Each grading item has a collection of ready-made phrases to be added to the feedback of that subcategory.

The lecturer can decide on different weightings on categories. The grade limits are also customizable resulting in quite free opportunities on how to use the tool. This requires a bit more effort on setting up the tool on behalf of the lecturer, but allows the tool to be used in many different courses - some possibly even unrelated to computer sciences.

4. DISCUSSION

The tool is useful in courses where there are several submission to be graded by multiple graders but with only small amount of submissions the benefits are mostly limited to the more comfortable grading process. Of course if the grading takes several days of time the tool might help the grader to stay consistent throughout the whole process.

The objectivity aspect of ALOHA is not that useful when an assignment is graded with the scale accepted/rejected especially if almost every student should pass the assignment. The feature that calculates the actual grade is obsolete, but ALOHA can still be used to calculate possible bonus points. Nevertheless, the tool can still be used to write good feedback texts.

ALOHA is currently developed and used only in programming courses, but its usage is actually not at all limited to programming nor even computer sciences. The limiting issue is that the building of a grading schema for an assignment requires moderate programming skills. Because it does not really have to be the teacher himself who does this, the tool could be used in other disciplines as well for example to grade project works or essays.

5. CONCLUSIONS

We have introduced few problems related to using multiple graders which is common in mass courses. ALOHA was built to easen these problems concerning consistency and objectivity in grading but also to make the grading process more comfortable for the grader resulting in proper feedback for the students.

The tool has been taken in use and it seems to make the grading process more convenient for the graders and also for the teacher. To test the consistency and objectivity, we will conduct a statistical analysis between the grading distribution of the graders using ALOHA to see if there are any differences.

6. ACKNOWLEDGMENTS

We would like to thank the rest of the group implementing the tool: Päivi Aho, Jarno Keskikangas, Harri Luoma and Hanna Sillanpää.

7. REFERENCES

[1] K. Ala-Mutka and H.-M. Järvinen. Assessment process for programming assignments. *Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT'04)*, 2004.

[2] K. M. Ala-Mutka. A survey of automated assessment approaches for programming assignments. *Computer Science Education*, 15(2):83–102, June 2005.

[3] K. Becker. Grading programming assignments using rubrics. In *ITiCSE '03: Proceedings of the 8th annual conference on Innovation and technology in computer science education*, pages 253–253, New York, NY, USA, 2003. ACM Press.

[4] M. Daniels, A. Berglund, A. Pears, and S. Fincher. Five myths of assessment. In *ACE '04: Proceedings of the sixth conference on Australasian computing education*, pages 57–61, Darlinghurst, Australia, Australia, 2004. Australian Computer Society, Inc.

[5] S. Habeshaw, G. Gibbs, and T. Habeshaw. *53 Problems With Large Classes*. Technical and Educational Services Ltd., Bristol, U.K., 1992.