Summary results of the Open Universiteit Nederland (OU)

Course

The Open Universiteit is a distance university with open registration, which means that there are no prior education requirements. The course Advanced Object Oriented Programming (5 ECTS) is the second programming course of the bachelor computer science and covers the more advanced OO programming concepts. The material is suitable for self-study. Five online lectures are given for additional support.

The topics covered are: inheritance including abstract classes and interfaces, generic and enumeration types, error types and debugging, exceptions and robust programming, concurrency, GUI programming and, finally MVC and the design pattern Observer.

After the topic robust programming, students make a hand-in assignment. This must be made sufficiently. The course ends with an exam. The exam grade is the final grade of the entire course.

The submission assignment takes 15 hours and concerns the design, implementation and testing of some functionalities of an agenda application. Each student's solution is accessed in detail and each student receives detailed feedback based on a rubric.

Baseline measurement

The baseline concerned the course as it has been given since 2018. The course ran from February 2022 to April 2022. About 80 students registered for the course.

Validation measurement

The validation concerned the course as it has been given in the period February 2023 to April 2023. Again, about 80 students registered for the course. The course has been adjusted compared to the baseline meeting: It is explained what contracts are and how to use them. Covered are the concepts method contract, precondition, postcondition, footprint, class invariant, and the semantics of a contract. Contracts are described in informal language. Students learn to use contracts, i.e. how to reason about program correctness and how to implement functional and test code step by step. Students is asked to write contracts for simple functionalities.

Rubric

A rubric is used for grading and feedback. The rubric distinguishes between an assignment-specific part and a general part. The assignment specific part contains the components domain modeling with UML class diagram notation, functionality happy path, functionality robustness, design with UML class diagram notation, code implementation, documentation, and testing. The (QPED) general part consists of the concepts: modularity, datatypes, readability, dry principle, flow, api documentation, correctness, robustness, test traceability and test completeness. A scale of 1 to 4 is used. The rubric is used during the baseline as well as the validation phase.

Questionnaire

The standard QPED questionnaire is used. Questions are about previous programming experience, self-efficacy, perception of the fundamental elements that define programming and software quality, programming habits (like the use of meaningful variable names, naming conventions and coding styles) and the use of tools (like style checkers and testing tools). The Questionnaire is used during the baseline as well as the validation phase.

Main results

Rubric baseline

36 Students were involved during the period of Feb-Apr'22. Students usually score a good level (3 on a scale 1 – 4) on readability, test traceability, flow and API documentation. Students score lower on test completeness, robustness and the choice of suitable data types (often score 2 on a scale of 1-4).

Rubric Validation

28 students were involved in the period of Feb-Apr'23. The aspects on which students had the most positive ratings (4) were flow, test traceability and data *type.* The aspects on which some of the students had the difficulty are API documentation and test completeness as there are shares of students who received the low scores of 1 and 2. However, even for these two aspects, shares of students with positive ratings of 3 and 4 prevailed.

Rubric comparison

The results obtained from 35 students in the baseline phase were compared with those of 28 students in the validation phase. Overall result is a positive redistribution of students across the various score ranges. In the baseline phase, a majority of students achieved scores of 3. Instead, in the validation phase, there is a decrease in the proportion of students obtaining scores of 2 and 3, while there is a significantly higher percentage of students attaining the highest score of 4. This pattern recurs in the majority of the items, in particular test traceability, test completeness robustness.

Univariate and bivariate analysis techniques were employed to compare the results, aiming to identify patterns of differences and similarities between the data collected in the two phases. Moreover, statistical tests such as Chi-square, Kolmogorov-Smirnov, U of Mann-Whitney, and Gamma tests were utilized for the detection of statistical equivalence, and difference between the two distributions of data and their respective central tendency (the median).

Questionnaire

In most of the items the students in the validation phase consider themselves to be more skilled than those in the baseline. These results suggest the positive impact of the learning tools implemented in the project on the skills acquired by the students.

Conclusion

This study shows a positive effect of the use of specifications on the quality of the functional code and completeness of the tests.

Detailed information can be found here.