ISSN:2184-044X ISBN:978-989-54312-6-7 © 2019
DOI: 10.36315/2019v2end075

EFFECTIVE PROGRESSION MANAGEMENT WITHIN VERY LARGE CLASSES IN COMPUTER SCIENCE EDUCATION

Paul Sage, Darryl Stewart, Philip Hanna, & Andrew McDowell
School of Electronics, Electrical Engineering and Computer Science,
Queen's University Belfast (*United Kingdom*)

Abstract

For some students, attaining the programming skills required to become an effective software engineer can be a difficult process. The initial steps taken in this journey are critical for success, but in all too many cases, lack of early engagement leads to a high attrition rate across associated education programs. Against the background of ever increasing class sizes, this work focuses on novice programmers enrolled on a software engineering degree program and considers how group activities, peer mentoring and self-assessment, can positively influence retention rate and performance.

Keywords: *Active research, computer science education.*

1. Introduction

Software engineering continues to be a popular choice for further and higher education students, driven largely by demands from the software industry and the well remunerated careers on offer. Associated degree programs are generally over-subscribed and will generally attract well qualified applicants. However, the subject can be viewed as difficult to get to grips with, particularly in the initial phase of study. This is evidenced by progression figures which (year on year) show that in excess of 20% of year 1 students do progress directly to year 2, which is recognised as a significant problem as indicated in the work of Matthews (2014) and O'Brien (2016).

There are a number of indicators that typically arise which can impact progression rates for novice software engineers: The ***material is inherently difficult***:- learning to write software programs requires strong mathematics, logic, problem-solving and language skills; the ***material is progressive***:- given the nature of the subject matter, each topic presented requires competence in previous topics, so missing lessons can present significant problems for the learner; ***learners can feel isolated***:- class friendship groups form naturally, but many learners lack the social skills to work effectively in a group context. Consequently, for some learners, difficulties in making a good start with this discipline, all too often spirals out of control leading to decreasing confidence/performance and ultimate disengagement.

Progression issues have been addressed across a number of studies through encouraging enhanced peer interaction. For example, Freeman (2014) considers how replacing more traditional teaching practices with active learning can significantly increase the success rate for completion of ICT and STEM courses. This is further supported by work of Carver (2007), which describes the results of an active research study investigating the use of peer/pair programming techniques (working in small groups) in teaching computer science and software engineering subjects. A significant goal of this study was to examine how retention could be improved and the reason for fluctuations, with the main conclusion indicating increased retention and a favorable student response.

However, issues with retention and progression remains a significant problem, further compounded by increasing enrolment figures; current statistics for Queen's University Belfast (QUB) indicate an intake on excess of 400 students across a range of associated degree programs, with class sizes in excess of 270 for individual programming modules. Classes of this size are difficult to manage and offer a poor staff/student ratio.

Introductory programming modules at QUB typically do not require prior knowledge or formal group work and students are individually assessed. Clearly, group/peer associations form an essential part of any career in this discipline but these activities seem to be widely ignored at this formative stage, and retention and progression problems remain. Consequently, based on the experiences obtained from an initial study, the main aim of this work is to look at how learner engagement and retention figures can be improved for novice programmers by encouraging early peer/group activities and learner reflection on progress. This work considers the suitability of introducing informal group associations to the first year programmers with a view towards assisting all learners with the course material. The application of peer mentoring and peer review techniques are considered, with particular emphasis on helping learners identified of being most at risk of achieving lower performance. The next section considers the background and results from this initial study conducted with smaller class groups and how this can be enhanced for significantly large classes.

2. Active research study and analysis

Using a smaller class group (of around 21 learners), an initial action research study was conducted to address progression issues. This class group was divided into teams of 4/5 members, where each group was encouraged to sit together in lessons, meet as a team outside lessons and mentor one another with course material and associated exercises. In addition, all teams were tasked with a group project, complementary to lesson material. Team members were encouraged to work collectively to solve given tasks and asked to periodically reflect on their progress/participation and the involvement/performance of peers. A randomised mechanism for team allocation proposed by McClelland (2012) was initially considered. However, to enable an effective initial baseline for the study, teams were composed to reflect a balanced range of abilities as informed by an initial formative assessment. To enable effective analysis of this work, evaluation criteria was bound to what can be usefully measured, in terms of quantitative and qualitative data:

a. ***Leaner Attendance***: This information is readily available from registers, with a positive outcome measurable through a sustained (or improved) attendance over the study period.
b. ***Team Performance***: As this study was conducted during normal course delivery, a number of (formative and summative) assessments were applied at different points prior to, during and after the action step, providing a useful indicator as to how groups/teams and individual learners responded to the intervention.
c. ***Peer Assessment***: Individual learners were asked to assess the performance of peers over the duration of the study, in terms of the contribution made, to provide a direct measurement of individual engagement and team cohesion.
d. ***Self-Assessment***: Learners new to programming, particularly those that indicate difficulties, typically demonstrate lower confidence in the material as a course progresses. Consequently, as each stage of the action step is delivered, learners were surveyed to obtain a ‘confidence measure’ on how they perceive their progress. This was applied at team/group level and at the individual level, where leaners participate in the collective team view and (in confidence) could express their personal views.
e. ***Impact on ‘At Risk’ Students***: Learners in this category were tracked as the study progressed to both encourage and facilitate peer support and/or tutor intervention as required.

2.1. Action step structure

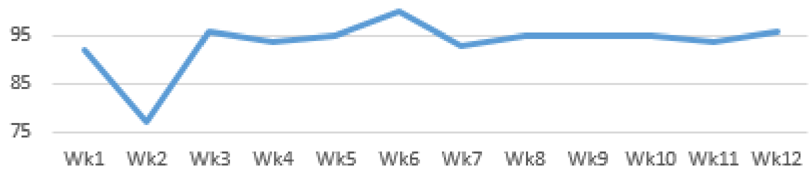
Content of the action step was chosen to reflect a semi-formal software engineering context, as informed by the work of Pope-Ruark (2011) and consisted of three elements of work (software design, implementation and testing) conducted over a three week period, with defined deliverables at the end of each phase. All teams were asked to participate in a brief quiz related to the given topic as each phase was completed. Teams were also surveyed to get a measure on perceived team performance and to reflect on individual performance. The main aim in monitoring team and individual perception of performance together with actual assessed performance, was to provide a better insight into the uptake of the material not just in terms of raw figures but also in terms of obtaining a realistic measure of learner ‘confidence’. As mentioned in the introduction, one of the difficulties associated with learning to program (and one of the main reasons for lack of progression or ‘dropping out’) is the perception of ‘being lost’. This is exacerbated by the progressive nature of the material, with attendance gaps magnifying the problem. These problems ultimately become visible through summative assessment but this (in many cases) is too late to rectify. In addition, learners are not good at coming forward directly with difficulties with the learning outcomes.

To facilitate a balanced study, the class group was divided into four teams of four learners, with an additional team of five learners. Team selection was based on performance in a class test (formative), given as a normal component of the programming module, with each group containing a level distribution of attainment overall. Given the study requirement of assisting learners considered to be most ‘at risk’, candidates in this category were identified using a combination of assessments scores and attendance figures to date.

2.2. Outcome and analysis

Attendance: The attendance for learners participating in the study is provided in figure 1 below. This gives a broad indication of engagement prior to, during and immediately after the action step.

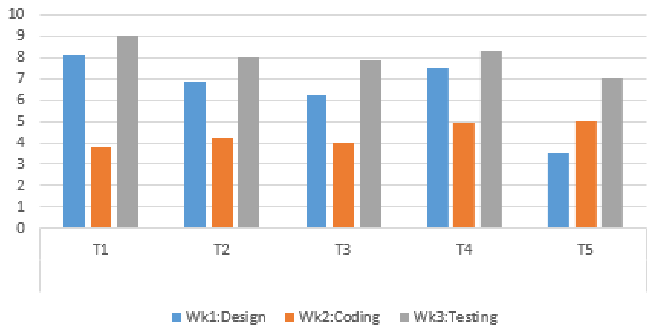
Figure 1. Attendance.



Attendance figures, while initially uneven, are more settled prior to and for the duration (weeks 8, 9 and 10) of the study and remained at a strong level of above 95%. On analysis, levels of attendance are considered to be excellent and were typically above other module attendance during the same period.

Team Performance: This data was collected at the end of each weekly cycle and represents team quiz and task submissions, together with collective team feedback as a (team) prediction of estimated performance. An ideal comparison would demonstrate a close alignment with actual and expected performance, representing an accurate perception of progress, irrespective of the level of actual attainment. Underestimation of performance (if severe) could indicate a lack of confidence to some degree. Some overestimation of performance could be considered as indicating team confidence. However, significant overestimation of performance could be considered more of an issue, indicating problems with the material and lack of engagement. It is important for individuals to have a realistic view of how well the team is functioning in terms of meeting the learning outcomes. A closer convergence between actual and perceived performance indicates that a team is able to identify problems (where they exist) more rapidly, thus building on confidence as work progresses. What follows is an analysis of each team's performance measured against a performance estimate, over the three week period of the action step.

Figure 2. Team Performance.



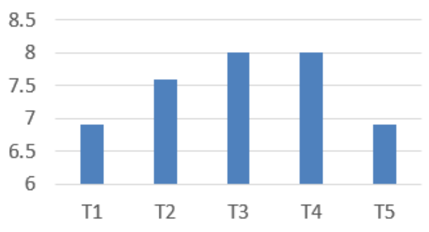
For two thirds of the study, team 1 performed better than their estimate, indicating some initial wobbles in confidence. However, as the study progressed, this seems to improve with a degree of convergence between actual and estimated performance. In consideration of performance as a whole, the graph indicates a rising trend, notwithstanding the dip in week 2 (which contains the most technically challenging work), demonstrating good progression with the material.

The comparative results for team 2 indicates some over estimation of performance across the study period. However, this converges somewhat by week 3 and the level of confidence remains high across the study. Actual performance is at an excellent level, although this dips a little in week 2 (as is the case for most teams), which is reflective of the level of task difficulty. Again, team 3 slightly over estimated across the study but does indicate a rise in confidence through time. Actual performance is generally in line with other teams and demonstrates a rising trend. Actual and estimated performance by week 3 is close, sitting at around 80%, indicative of a team that is both confident and capable.

Team 4 demonstrated a very similar performance profile to the other teams, with an overall rising trend. Overall performance is excellent with actual and estimated performance converging nicely by week 3. Results indicate that (again) this team is functioning well together to produce an effective outcome. The profile for team 5 differs a little from teams 1 to 4. Results indicate a linear rising trend across the study, with a slight overestimation for each task. Overall performance is strong by the end of the study and demonstrates a significant growth in both confidence and ability, from some difficulties in week 1 to sound performance in week 3.

Peer Assessment: One important part of the study was to consider how team peers viewed each other's contribution. Consequently, teams members were asked to rate their peers on a scale of 1 (low contribution) to 10 (high contribution). The peer assessments for each individual were averaged to give a rating per team member. These ratings were then averaged for each team and is presented in figure 3 below.

Figure 3. Peer Assessment.



For all teams, results indicated a good degree of cohesion. In particular, teams 2, 3 and 4 appeared to be most comfortable together, as evidenced by the peer assessments and by observations within the classroom.

Self-Assessment: Results indicated a rising trend in both confidence and actual performance with the task material. The team confidence measure was supplemented by individual confidence (self-assessment) survey, taken privately to allow learners to express their thoughts aside from the collective team view. This data, averaged across the three week study period across the whole class group, indicates a general rising trend, in line with broad team perceptions.

At Risk Learners: On establishing team membership prior to commencement of the study, a number of learners (A, B, C and D) at most risk of disengaging (or 'dropping out') were identified, using a combination of initial and attendance data. In addition to data generated through group activities, the results of the first summative (interim) assessment for the programming module were brought into consideration to review the overall impact of the study. Three individual learner measurements (pre-study assessment, interim assessment and peer assessment) are presented for comparison in Figure 4 below.

Figure 4. Assessment Profile: Learners at Risk.

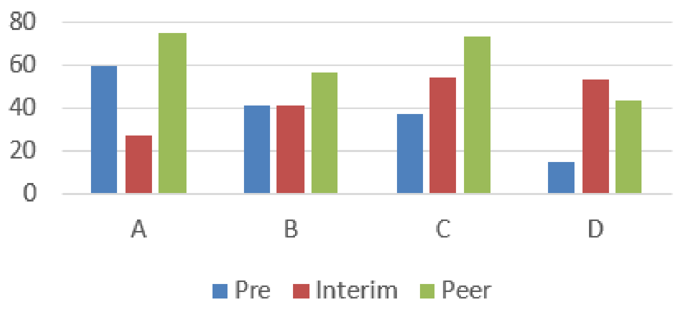
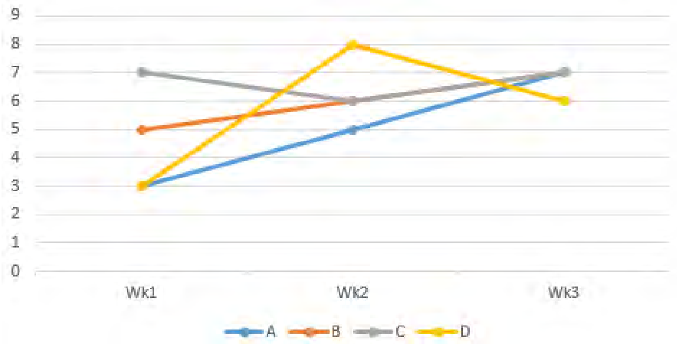


Figure 5. Confidence Measure: Learners at Risk.



On analysis, learner A performed fairly well in the initial assessment but dipped significantly in the second. However, engagement as perceived by peers was at a high level and in line overall team performance backed up by the confidence measures, which indicated a rising trend. Learner B achieved a slight improvement between the first and second assessments. This, coupled with a good peer assessment from the associated team and a rising confidence measure, demonstrates a positive impact for the intervention. For learner C, there is a sustained increase across the three measurements, with a strong indication of team contribution. For learner D, an improvement in assessment performance is evident, however, peer opinion indicates a moderate level of engagement, which could be improved upon. This could perhaps be linked to an initial rise, followed by a fall in confidence and warrants further investigation.

3. Conclusion and further work

By and large, individuals were very willing to engage with their respective teams. This became apparent in the competitive nature of team behaviour and by the lack of tutor input required to achieve a good outcome. Teams formed natural focus groups and genuinely relished the opportunity to influence the study and working with peers was very well received. During the study, it was clearly evident that learners considered 'at risk' of falling behind, were not allowed to lapse; perhaps this can be attributed to engendered team ethos and the desire 'not to let the side down'. Feedback from students involved with the initial study was very positive. Moreover, an element of 'sporting challenge' between teams has produced good results and (without encouragement) they have continued to support one another well beyond the scope of the study. More importantly, 3 out of 4 students initially identified as 'at risk' successfully completed the module, with overall results significantly above other class groups external to the initial study.

These results have been very encouraging. Consequently, an enhanced study is currently underway, encompassing a much larger class group of 270 learners currently taking a year-long (2 semester) introductory programming course, with lab exercises and weekly formative assessments as an integral component. In this new study, learners are divided into groups of 10 and are required to sit at the same bench for each weekly session. At time of writing (December 2018), the first semester has been completed. During this time, learners have been working individually on course material and formative assessments and while they have been encouraged to assist one another, there has been no formal requirement to work as a group. Learners have been regularly surveyed, asking each to assess weekly progress/performance with the material (as a measure of confidence). This has provided a rich data set (in conjunction with summative assessment results), forming a baseline for continued work in semester 2.

Going forward, learners will be actively encouraged to work on small group projects. It is envisaged that three such projects will be examined (each of a three week duration) with a restructuring of individual groups at the end of each period to normalize performance across the entire group. This would avoid formation of new 'comfort zones' and should encourage sustained engagement (and hopefully an increase in performance to match).

In line with the previous study, the performance (both actual and self-assessed) of learners considered to be at risk will continue to be monitored. Results will be published, post analysis at the end of semester 2 in this academic cycle.

References

Carver, J. (2007). Increased Retention of Early Computer Science and Software Engineering Students Using Pair Programming. *Software Engineering Education & Training 20th Conference on Software Engineering Education & Training (CSEET'07), IEEE*. 1 (3), 302-317.

Freeman, S. (2014). Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences*. 11 (2), p34-40.

Hunter, A. (2009). High-tech Rascality: Asperger's Syndrome, Hackers, Geeks, and Personality Types in the ICT Industry. *New Zealand Sociology*. 24 (2), p34-56.

Matthews, D. (2014). *Dropout rate tumbles, but not among IT crowd.* Retrieved Nov 11th 2016, from https://www.timeshighereducation.com/news/dropout-rate-tumbles-but-not-among-it-crowd/2014857.article

McClelland, P. (2012). The influence of randomly allocated group membership when developing student task work and team work capabilities. *Journal of Further and Higher Education*. 36 (3), p351–369.

O'Brien, E. (2016). *Why are computer science drop-out rates so high?*. Retrieved Nov 11th 2016, from http://trinitynews.ie/why-are-computer-science-drop-out-rates-so-high/

Pope-Ruark, R. (2011). Let's Scrum: How Scrum Methodology Encourages Students to View Themselves as Collaborators. *Teaching and Learning Together in Higher Education*. 3 (1), p1-14.