A Programming Crisis? : An study based on the downfalls of modern programming practices and learning platforms and their effects on students in order to aid in the creation of an innovative and interactive web-based learning application to motivate learning.

Trevor Kiggundu: 001001720

October 2019

BEng Software Engineering

A project proposal submitted in fulfilment of the requirements for the module, Final Year Project, Computing and Information Systems Department, University of Greenwich.

1 Introduction

The current society is a “technology-based” (Susskind, 2015) one, and there seems to be no sign of that stopping any time soon. The demand for these types of technology professionals is at an all-time high. A career in this profession is also one of the rarer ones in which the number of job opportunities continues to increase despite the parallel increase in recruits available. Or at least that is what it seems like on the outside. Astonishingly, a large percentage of job applicants, even “those with masters' degrees and PhDs in computer science, fail during interviews when asked to carry out basic programming tasks” (Kegel, 2014). This points to a problem regarding the external motivation of junior programmers, as “most programming courses don't provide the kind of experience that real programming gives” (Kegel, 2014), only laying down a good base for the student to explore and learn more themselves. Without the proper motivation, and the correct learning tools, it is difficult for young programmers to possess both the theoretical and practical skills needed to work in the real world, as too many graduates only possess the former as opposed to the latter.

2 Problem Domain

The main focus of this project is to create an innovative and interactive web-based learning application to motivate students to learn programming. Designing an “effective” and “interactive learning environment requires understanding the intricate relationships among people” (Repenning, 2006), so there will need to be more background research on learning practices. In order to do this, a secondary objective will be introduced, which is to research and investigate the motivating factors that students have regarding the task of learning to program. The main investigation will include conducting deeper research into more innovative UX and UI designs, and HCI factors in order to provide an alternative to the various already available online websites that aim to teach programming through web applications. These applications, however, are still important, as they provide a different learning approach through games, quizzes and “visualization features to improve students’ understanding of the programming concepts” (Moons et al., 2013). The bulk of these investigations will be documented in the literature review, and the web application will be created in lieu of these findings.

2.1 Further Research Ideas

Future research regarding the motivation of students to learn programming could venture into exploring the reasons that students choose to pursue this job path today. Technological advancements over the years mean that the range of topics that fall underneath the tech field are vast, from games design to industry software development. The diversity in the age and gender of those involved in the field has also been noted, with women "building careers as professional engineers in a high technology industrial organization" (Evetts, 1998) more frequently than ever. Individuals are also picking up the practice of programming at an even younger age than before, mostly through online games and reasoning programs, as these seem to be promising tools for "assessing computational thinking in young programmers" (Touretzky et al., 2017). A study could be done on whether these young learners are being guided/persuaded into joining the field or are just inquisitive enough and have completely intrinsic intentions towards learning to program.

3 Methodology

3.1 Background Research

Various methods have been used to teach programming; some are more outdated and/or successful than others, but all were created to achieve the same goal. The conventional teacher-student model is the oldest and most common. This model is quite self-explanatory, though it does branch out into practices such as apprenticeship, more commonly known as 'placement', where the student is offered "continuous feedback" by one or more mentors to not only differ from "traditionally constructed courses", but also offer the "most efficient means of learning" (Vihavainen et al., 2011). Another form of learning is self-study, in which the student teaches themselves the required material using various means (ex: books, online websites, games, journals etc.). A study by Bergin et al. (2005) sheds a brighter view on self-regulated learning, stating a direct link between intrinsic motivation and SRL (Student Regulated Learning), with students exhibiting such behaviors being more inclined to "perform better in programming" than those with more extrusive motivational factors. Learning via web applications best fits this category, as it requires the student to explore

more about the given subject in their own time. Detailed below are the various objectives and methodologies that will be implemented in the proposed project.

3.2 SMART Objectives

Smart Objective 1:

- Within the allocated time-period (1.5 months), aim to collect user feedback regarding the usefulness of the learning application created at the half-way level. User feedback will also be used to determine which topic will be taught on the application. The feedback will be anonymous and will be directed accordingly to create the least amount of bias possible for more reliable data collection.

Smart Objective 2:

- Within the allocated time-period (1.5 months), plan to create a web-based application prototype to be both client and developer tested. This will allow for early feedback so that the appropriate changes can be made before the final product is presented.

Smart Objective 3:

- Within the allocated time-period (6 months), aim to create a fully functional learning-based web application with more interactive features than the prototype. The final product will be white and black box tested, as well as client and developer tested again. Feedback will be taken from this final product and compared with the prototype.

Smart Objective 4:

- Within the allocated time-period (6 months), aim to compile all the research, data and results collected in order to write a completed research paper, poster and presentation in fulfilment of the requirements for the module and final year project. This will be subject to the success of the web application created, as well as the user feedback.

3.3 Deliverables

The proposed project will deliver two specific products; an innovative and interactive web application in conjunction with a project report written in LaTex. The latter will be a direct product of technical research, literary analysis, and the findings documented from the study conducted.

3.4 Project Methodology

As mentioned before, the proposed project will not only include investigating the motivating factors regarding learning programming, but also the user interface designs of the already available learning avenues. This will help to develop a web application that aims to provide an alternate learning experience than the traditional lecture and tutorial model used by many learning organizations. This is important, as “traditional ways of assignment grading are not scalable” and often do not “give timely or interactive feedback to students” (Tillman et al., 2013). The application itself will be created to teach a programming language/topic to first and/or second year students that has not been chosen yet; this will be handled in either one or two ways. A survey via questionnaire will be uploaded online for students themselves to choose which topic to learn about. Alternatively, the web application will be catered to teach a subject that is already being taught by lecturers at the university, as this will provide a sample size, at most, as large as 250-300 students, provided they all respond to the survey. The final goal indeed is to keep learners motivated, so the web application will have different interactive features such as quizzes, checkpoints and level progression to keep the user interested in the subject. These will aim to keep the user more interested, as “improving learning effectiveness has always been a constant challenge” and “computer games are one means to encourage learners to learn” (Long, 2017). Previous experience from the Web Application Development, Database Technologies and Application Development modules will be used to develop this application.

3.5 Systems Development Methodology

This section aims to detail the systems development methodologies considered during the planning of the proposed project. The Extreme Programming and Spiral models were eliminated immediately due to the nature of the task at hand. The Waterfall model was also eliminated as it is a model that does not promote user interactivity until the end of the project

cycle, a big staple in the production of this application. It is also aimed at large and expensive projects, two variables that do not describe the proposed idea. The system development methodology that would best suit this project is a close call between the Agile Development model and the Rapid Application Development (RAD). Both models are low costing, relatively defined and informal, and have a “small-to-medium scale and of short duration” (Kyriakidou, 2018). However, the negatives of the agile development model centered around the fact that it involves customer collaboration at a much higher level than the RAD model. Given the time constraints, user interactivity and complexity of the project, the conclusion was made that the best systems development model to follow would be the Rapid Action Development (RAD).

3.6 Test Bed

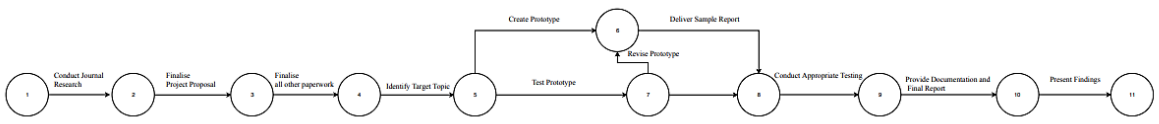
A series of white and black box tests, as well as unit tests, will be run to test the functionality of the web application, database integration, HCI factors and the user documented features of the project. The testing will be run on both developer and user side to eliminate any possible bias and receive as much feedback as possible.

3.7 Models

Models to be implemented will include UML diagrams, class diagrams and updated Pert/Gantt charts. Functional and non-functional requirements as well as ERD diagrams will also be included.

3.7.1 Pert Chart

Shown below is a pert chart detailing some of the objectives that are to be achieved during the duration of the proposed project.



4 Evaluation

Data collection will be an instrumental part of the process, as the proposed project attempts to solve real world programming issues. Previous project proposals were halted due to the difficulties that accompanied the task of finding reliable and readily available test data. This proposal will be different; as year 1 and 2 students will provide data about various topics such as interest in programming, desired topics, and general user interface feedback. The appropriate legal, social, ethical and professional guidelines will be implemented throughout the study. After the web application is created, user feedback will be collected from a select number of participants to gather information on whether the application has had any positive/negative impact on the concept of the selected topic, and if it has increased confidence/motivation in the individual. This is very important, as students can find the process of learning to program to be a "difficult and even unpleasant task" without the proper "guidance and feedback" (Watson et al., 2011) given back to them. The completion of these objectives, as well as the ones documented in the 'problem domain' and 'project methodology' sections, will determine whether the proposed project will be successful.

5 References

- Bergin, S., Reilly, R. and Traynor, D. (2005) Examining the role of self-regulated learning on introductory programming performance, *Proceedings of the 2005 international workshop on Computing education research - ICER 05*.
- Evetts, J. (1998) Managing the technology but not the organization: women and career in engineering, *Women in Management Review*, 13(8), pp. 283–290.
- Kegel, D. (2014) How To Get Hired -- What CS Students Need to Know, *How To Get Hired -- What CS Students Need to Know*, Kegel, [online] Available at: http://www.kegel.com/academy/getting-hired.html (Accessed October 27, 2019).

- Kyriakidou, A. (2018) COMP1632: Systems Development Project, *COMP1632: Systems Development Project*, London.
- Long, J. (2007) Just For Fun: Using Programming Games in Software Programming Training and Education, *Journal of Information Technology Education: Research*, 6, pp. 279–290.
- Moons, J. and Backer, C. D. (2013) The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism, *Computers & Education*, 60(1), pp. 368–384.
- Repenning, A. (2006) Programming Substrates to Create Interactive Learning Environments, *Interactive Learning Environments*, 4(1), pp. 045–074.
- Susskind, R. E. and Susskind, D. (2017) *The future of the professions: how technology will transform the work of human experts*, Oxford, United Kingdom, Oxford University Press.
- Tillmann, N., Halleux, J. D., Xie, T., Gulwani, S. and Bishop, J. (2013) Teaching and learning programming and software engineering via interactive gaming, *2013 35th International Conference on Software Engineering (ICSE)*.
- Touretzky, D. S., Gardner-Mccune, C. and Aggarwal, A. (2017) Semantic Reasoning in Young Programmers, *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education - SIGCSE 17*.
- Vihavainen, A., Paksula, M. and Luukkainen, M. (2011) Extreme apprenticeship method in teaching programming for beginners, *Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE 11*.
- Watson, C., Li, F. W. B. and Lau, R. W. H. (2011) Learning Programming Languages through Corrective Feedback and Concept Visualisation, *Advances in Web-Based Learning - ICWL 2011 Lecture Notes in Computer Science*, pp. 11–20.