2022

PREPARING IS PROGRAMMING STUDENTS FOR WORK BY ENHANCING A CONNECTIVIST TEACHING APPROACH BY SCAFFOLDING PRACTICES

Machdel Matthee
University of Pretoria, machdel.matthee@up.ac.za

Phil van Deventer
University of Pretoria, phil.vandeventer@up.ac.za

PREPARING IS PROGRAMMING STUDENTS FOR WORK BY ENHANCING A CONNECTIVIST TEACHING APPROACH BY SCAFFOLDING PRACTICES

Machdel Matthee and Phil van Deventer
Department of Informatics
University of Pretoria
Machdel.matthee@up.ac.za; phil.vandeventer@up.ac.za

Abstract:

Programming modules often do not use prescribed textbooks due to the rapidly changing content. Also, in practice, software developers draw on online communities and online resources to learn. This approach shows similarities to a learning theory called connectivism. Connectivism posits that knowledge is to be found in networked relationships implying that learning concerns the skill to navigate and create those networks. Connectivism is therefore considered an appropriate teaching approach to prepare IS programming students for the world of work. However, this is not possible without guidance from lecturers. This research-in-progress reports on how scaffolding practices supplement connectivist strategies in a second-year IS programming course. The hypothesis is that these interventions will have a positive effect on students' programming self-efficacy and, therefore, their learning experience. Preliminary results show that students appreciate the interventions and have greater self-confidence in tackling more complex programming projects.

Keywords: IS programming students, programming self-effficacy, connectivism, scaffolding, cognitive load

I. INTRODUCTION

Programming modules often do not use prescribed textbooks due to the rapidly changing content. Also, software developers in practice, draw on online communities and online resources to learn from. For these reasons, the current approach in teaching a second-year programming IS course (INF272) at the University of Pretoria focuses primarily on autonomy and self-management of learners, online communities (in the form of WhatsApp groups) and diversity of available resources on the internet. For example, we use slides compiled from different online resources and point students to diverse programming tutorial websites. However, this approach (which has elements of what is known as a connectivist teaching methodology) is experienced as overwhelming, especially for beginner programmers, as many of these students find it difficult to identify examples on the appropriate level.

Feedback from the 2021 group of INF272 students underlines this: "*Provide more examples", "Give us fake tests to test comprehension", "Give us more activities to do so that we enhance our knowledge", "Provide more examples to display the work."*, "*More examples to practice.", and "How do we know what we need to know?".*

Although a connectivist approach provides many advantages and ties in with contemporary programming practice, it is hamstrung by the fact that it does not give enough attention an aspects of learning which have been highlighted by cognitive learning scientists as crucial in successful teaching and learning. These are the limitations of human working memory, which in turn necessitates proper scaffolding of complex and new learning material. Scaffolding refers to the simplification of a task into smaller units, with support just above the level of the learner. Taking this into account will make it easier for students to acquire the proper fundamental knowledge that will enable a larger cohort of students to reap maximum benefit from a connectivist approach to teaching and learning.

Students have access to recorded and live lectures and comprehensive and challenging homework assignments. However, this has proven to not be enough within the current approach.

However, we believe their learning, and therefore self-efficacy will improve if the shortcomings in the current approach are addressed. For example, provide students with access to worked examples and a pathway of exercises compiled from existing resources. These examples can then be integrated into the presentation of the course adhering to the principle of scaffolding, demonstration of best practice and thereby limiting students' cognitive load - which is a result of the limitations of working memory.

The objective of this research project is therefore twofold: 1) to establish scaffolding to students, through meaningful micro-learning practices (such as worked examples and short exercises)./ This will equip them with fundamental knowledge and a road map to better navigate and choose from the vast number of internet resources available. 2) to investigate the effect of this intervention on the students' self-efficacy and learning experience.

II. LITERATURE STUDY

Connectivism

The approach in the second-year module described above is an example of the application of connectivism, a recent learning theory. Connectivism considers the learning process as taking place through social networks enabling the combination of different ideas and resources. The central role of digital technology is acknowledged, as it provides new choices for learning. [WGU, 2021; Downes, 2019]. Connectivism principles for successful learning are autonomy, diversity, openness and interactivity [Al Dahdoud, 2018].

Downes [2019], in his useful overview of research on connectivism, highlights the successes and advantages of applying this learning theory. The positive effect on motivation and self-efficacy is well documented in research [Downes, 2019]. Transue [2013] considers effective teaching in such a learning environment as guiding the identification, navigation and evaluation of information available in these networks. However, there are researchers [Clara and Barbera [2013] in Downes [2019]] who express their concern about the failure of connectivism to explain concept formation. Al Dahdouh [2018] comments on the vagueness of connectivism regarding how connections are formed and what criteria are used for those choices. From observing learners, he suggests a node-connection process: "1) planning and forethought; 2) cognitive processing and 3) evaluating". Planning and forethought suppose already formed concepts often in the form of foundational knowledge. As aptly put by Kirschner and Hendrick [2020], "what you know determines what you learn".

Denning et al. [2021:31] argue that by focusing only on 'writing code' in programming courses, students do not get an appreciation for the "thinking and design practices for the real world of large programs, systems, networks, and user communities. It does not describe how computing professionals meet concerns, answer threats, approach problems, rise to opportunities, or communicate practices." For this reason, despite its shortcomings, we believe that connectivism is the appropriate strategy to prepare IS programming students for their future careers. It is also evident that students can only choose from the vast sea of information and make meaningful connections if they have at least some existing mental schemas to inform the choices. Therefore, we suggest adding scaffolding practices (including micro-learning and worked examples) to the current pedagogy.

Scaffolding

Cognitive load theory "is based on the limited ability of the working memory to code information" [Kirschner and Hendrick, 2020:14]. In contrast to working memory, long term memory is almost infinite. Once a student has mastered domain-specific information s/he can access it without limitations. Cognitive load theory encourages teachers to assist students in building up and mastering schemata of domain-specific knowledge so that they are better able to solve problems,

rather than providing students with a vast amount of new information and expecting them to navigate and master this all at once. Scaffolding knowledge is one way of doing this.

Scaffolding refers to the support provided during a learning task by reducing the number of possibilities. The result is often a simplification of the task into smaller units, but the support has to be just above the level of the learner. The learner needs to understand each task's objective and how these objectives feed into the final objective. The scaffolding must be decreased as the learner's skills increase. [Kirschner et al., 2020].

Micro-learning refers to learning taking place in short intervals covering small chunks of content. This relates to connectivism as it focuses on "the development of the ability to form links between many ideas, each with each other and with different sources of information." [De Gagne et al. [2018] in Downes, 2019:119]. Micro-learning is often used in scaffolding practices as it presents smaller chunks of tasks – ideal for incremental levels of complexity.

Worked examples are exercises with complete solutions. Recent research on the value of worked examples by van Harsel, Hoogerheide, Verkoeijen and van Gog [2020] shows that integrating these in the teaching offering is more effective than only focusing on problem-solving. This approach can be combined with scaffolding by reducing the 'number' of solution steps in subsequent examples provided until a student has to complete the full exercise independently. This series of worked examples can be supplied in micro-learning chunks and as low-stake formative assessments (where students are provided a safe environment to try, make mistakes and learn).

Programming self-efficacy

Bandura [1977] developed self-efficacy as a theory of behavior change. It is defined as a person's belief in their own ability to execute their behavior to get the desired results. Ramalingam and Wiedenbeck (1998) applied this theory to the programming domain to create a measurement instrument for programming self-efficacy. Programming self-efficacy is defined as "PSE is the belief of one's ability to accomplish all the tasks related to programming, including simple and complex programming tasks" [Kittur, 2020:217]. Over the years, different PSE measurement instruments have been developed for different populations [Ramalingam and Wiedenbeck, 1998; Kukul et al., 2017].

The scale developed by Kittur [2020] uses four subscales: 1) students' self-efficacy relating to basic programming skills, 2) complex programming skills, 3) dependence on assistance and 4) self-regulation. Kittur uses this instrument to determine the relationship between student demographics and programming self-efficacy of electrical and electronic engineering students. This instrument is of interest to our study as the population for whom it was developed, has similarities to the Information System students. Similar to engineering students, the Information Systems students' focus is not on programming per se. Information Systems students are typically prepared to become business analysts in organisations, not software developers. (A small percentage of these students still choose to become developers, database administrators or fulfil other technical roles). That said, programming remains an integral part of the offering but is done with this focus.

III. THE TEACHING INTERVENTION

Overview of the course

The programming course is a second-year Information Systems programming year module course compulsory for the BCom Informatics/Information systems degree. Students need this knowledge to complete their comprehensive capstone project in the third year. The module outcomes are as follows:

After completion of this module, a student should be able to:

- MO1: Develop web-based applications using the Model-View-Control design pattern to meet a given set of systems requirements.
- MO2: Use object-oriented programming techniques and concepts to model a given scenario.
- MO3: Integrate database techniques and concepts in a software solution.
- MO4: Implement standardised data exchange tools to integrate web services in a software solution.
- MO5: Use best practices and debugging to improve the quality of a software solution.

The software and programming languages used are as follows: JavaScript, CSS, HTML, JQuery, C#, SQL server management studio and Visual studio 2019 (as IDE).

The lecture mode in 2021 and the first half of 2022 was entirely online. The teaching approach in 2021 entailed the following: Each week, students attended a live, online lecture of 3 hours where the new content was illustrated in a comprehensive exercise. After this, students attended a shorter practical session where some of the content was further elaborated. Before attending the lecture, students did a quiz on the work to be covered that week. In addition, they had seven homework assignments spread across the year and three semester tests, of which the two best were used for the semester mark. Students also belonged to different self-initiated WhatsApp groups.

The suggested teaching approach

In 2022 the suggested, altered teaching approach was piloted. It differs from 2021 in the following ways: The 3-hour live lecture was replaced by live online lectures illustrating smaller chunks of worked examples. The preparation quiz is still implemented, and only 6 homework assignments and 2 semester tests are needed. In addition, we included shorter practical assignments on increasing complexity building towards the homework assignment. These practical assignments are done in the labs with student assistants available.

Table 1: The suggested teaching approach

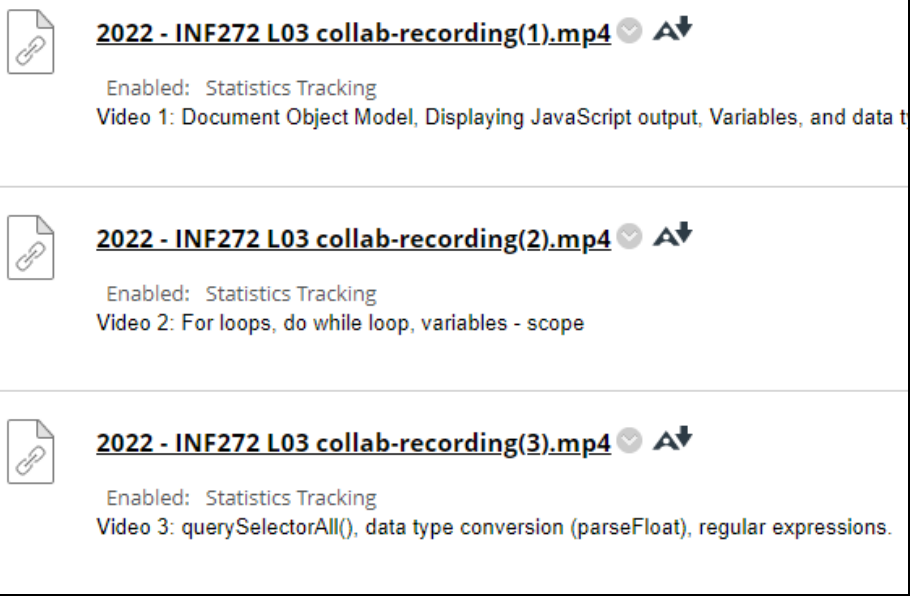
Teaching practice	**Time**	**Remarks**
Quiz	Weekly, in preparation for lecture	online
Study notes in the form of Powerpoint slides	Weekly	Self-study
Links to useful internet resources	Weekly	
Live lectures which provide worked examples. These lectures are recorded and made available to students.	Weekly	Online guided sessions that is broken down into small snippet activities that students complete along with the lecturers. (micro-learning)
Short exercises in the form of practical assignments. These exercises are short and increase in level of complexity towards the homework assignment	Weekly	Increasing in level of complexity and building towards the homework exercise (micro-learning and scaffolding) – in the labs.

Comprehensive homework assignment	6 per year	
Semester test – consisting of three questions of 10 marks each	2 per year	Practical tests to be completed in the labs. Projects are provided with some components missing. Students typically only need to upload those components – e.g. a .cs or .cshtml file etc.
Exam	1 per year	With a similar format as the semester tests.

Putting the ideas into practice

The following table provides a few examples of how our scaffolding practices supplement connectivist elements in the INF272 module during 2022.

Table 2: Putting scaffolding and connectivism into practice

Scaffolding practice to support connectivism	**Screenshots from the LMS, practical exercises and study guide**
Provide link to online worked examples and highlight the important ones	Work through the basic exercises on https://www.w3resource.com/javascript-exercises/javascript-basic-exercises.php (at least exercises 1,6,10, 27,30,41, 54, 95). There is no need to submit anything for this exercise (nr 3). Note that most of these exercises use functions. It was done in INF164 and we will focus more on it during lecture 4 as well.
Divide the online videos of online lectures into smaller meaningful chunk	

<table>
<tr>
<td>Explaining the concept of micro-teaching in the study guide</td>
<td>
2. In the live sessions students will be introduced to guided exercises and activities. These activities will be completed in small manageable chunks so that students can keep up.
3. As part of these live sessions, students will be given small “micro” practical activities that are linked to the larger “macro” homework assignments.
4. The relationship between “micro” activities and “macro” homework assignments are as follows:
This is a simplified example
<table>
<tr><td>Micro Practical Activity 1 (New skills from session 1)</td><td rowspan="5">Macro Activity (Homework Assignment) that combines the skills acquired from for example Activity 1 to Activity 5.</td></tr>
<tr><td>Micro Practical Activity 2 (New skills from session 2)</td></tr>
<tr><td>Micro Practical Activity 3 (New skills from session 3)</td></tr>
<tr><td>Micro Practical Activity 4 (New skills from session 4)</td></tr>
<tr><td>Micro Practical Activity 5 (New skills from session 5)</td></tr>
</table>
Number of micro activities per macro activity will vary.
5. If a student skips out on completing any of the Micro Practical Activities, then they will not develop the necessary skills to complete the Larger Macro Assignments.
</td>
</tr>
<tr>
<td>Get students to start using GitHub in order to prepare them for collaboration and an online community of programmers</td>
<td>
HW3 Submission
Posted on: Wednesday, June 1, 2022 1:20:57 PM SAST
Dear INF 272,
Please ensure that you have uploaded your HW3 assignment by using GitHub (follow the upload instructions posted if you are unsur
This is very important for us to mark your assignment and for you to build a portfolio that you could use in the future.
</td>
</tr>
</table>

To summarise:

Students are urged to consult online material by providing them with links to online resources (connectivism). To make the volume of information less overwhelming, students are guided towards the most important aspects to consider (scaffolding). Students have to create a GitHub account where they upload their assignments. This forces them to start using and exploring collaborative spaces for programming (connectivism).

The concepts of scaffolding and micro-learning are introduced in the study guide. Micro-learning is implemented by dividing the live lecture (available as videos aftewarads) into smaller meaningful chunks. Scaffolding is implemented by providing weekly short exercises. These exercises increase every week in complexity and contribute towards understanding the concepts needed to do a comprehensive homework exercise.

IV. RESEARCH STRAGEGY

Case study research will be performed on the INF272 module in 2023. We expect more or less 350 students to enroll for the course in 2023. This project aims to investigate the effect this intervention will have on the students' self-efficacy and learning outcomes. The hypothesis is that using scaffolding to supplement a connectivist teaching approach will increase students' programming self-efficacy, leading to improved learning. Self-efficacy will be tested using the complete measurement tool of Kittur [2020]. The self-efficacy questionnaire will be administrered in the beginning of the academic year and again at the end of the academic year – using the same group of students. The pre- and post- results will be compared to determine the effect of the intervention on self-efficacy. The marks of assignments, tests and exams of 2023 will be

compare to those of 2022 and earlier years. It is acknowledged that these comparisons are done on different groups but it may provide some indication of the effect of the intervention. The suggested teaching approach provided in table 1 was piloted in 2022. We also piloted the self-efficacy questionnaire. Table 3 provides the data collection and analysis strategies to be followed in 2023.

Table 3: Data collection and analysis strategies

Data collection	Data analysis	When	Purpose
Semester test	Descriptive and inferential stats	2x per year	To determine students' level of understanding and compare it with previous year's test marks
Self-efficacy questionnaire (see section 5)	Descriptive and inferential stats	At the start of the module and just before the final exam.	To determine students' self-efficacy levels.
Open-ended questions on the course, administered by the university – lecturer feedback survey	Thematic analysis	End of semester 1 and end of semester 2	To determine levels of satisfaction with the approach (comparing it with previous years' results) and get ideas for improvement
Homework marks	Descriptive and inferential stats	6x per year	To determine students' level of understanding and compare it with previous year's homework marks

Ethical considerations

All students enrolled in INF272 assessment marks will be used and asked to complete the self-efficacy questionnaire. It will be made clear that the data will be used to determine the effect of implementing the improved teaching practices. We are interested in the impact on their programming self-efficacy and marks. The questionnaire is anonymous, and participation is voluntary. We received ethical clearance in August 2022 and piloted the questionnaire with the current group of students at the end of August 2022. Unfortunately, this data cannot be used to determine the effect of our teaching intervention as we could not collect the data at the beginning of 2022.

V. PRELIMINARY FINDINGS

As mentioned above, the self-efficacy questionnaire was not approved in time to start this experiment in 2022. However, we could use the qualitative data obtained from the lecturer evaluation survey conducted by the university.

The feedback points towards an appreciation of the micro-learning and scaffolding practices: : "*the weekly practical and prep quiz is extremely useful especially when it is time to face the homework"; "Micro activities help with muscle building for the macro activities."; "Micro activities helped prepare me for homework assignments"; "The practicals really help in providing practice work for the module and make the homework assignment a bit easier to handle."*

In addition, the average of the first-semester test in 2022 increased by 20% from 2021. Although anecdotal, these are positive results. However, we still do not know the intervention's effect on the students' programming self-efficacy, but we hope to determine that in 2023.

VI. CONCLUSION

We showed that the complexity of the existing development frameworks, different aspects of Web development and the vast online information seem overwhelming to the programming students. We argued that the best way to teach the complexity of the network of tools and skills is through connectivism but that students need more structure and guidance.

This paper reports on research-in-progress that provides such structure and guidance for programming students. It is done by enhancing a connectivist teaching approach by incorporating scaffolding practices. The idea is to reduce students' cognitive load and assist them in building mental schemes to help them identify, evaluate and use online resources and communities for learning.

We believe our approach will improve students' motivation and self-efficacy, which have been shown to influence student learning positively. This, in turn, will increase throughput and help students to make a meaningful contribution to their workplace. Although anecdotal, preliminary results are positive.

VII. REFERENCES

Al Dahdouh, A. A. (2018). Jumping from one resource to another: how do students navigate learning networks*? International Journal of Educational Technology in Higher Education*, 15(45).

Bandura, A. (1977). Self-efficacy: toward a unifying theory of behavioral change. *Psychological review*, *84*(2), 191.

Denning, P. J., and M. Tedre (2021). Computational thinking for professionals. *Communications of the ACM*, *64*(12), 30-33.

Downes, S. (2019). Recent work in connectivism. *European Journal of Open, Distance and E-Learning (EURODL)*, *22*(2), 113-132.

Kirschner, P. A., and C. Hendrick. (2020). *How learning happens: Seminal works in educational psychology and what they mean in practice*. Routledge.

Kittur, J., 2020. Measuring the programming self-efficacy of Electrical and Electronics Engineering students. *IEEE Transactions on Education*, *63*(3), pp.216-223.

Kukul, V., S. Gökçearslan and M.S. Günbatar (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. *Eğitim Teknolojisi Kuram ve Uygulama*, *7*(1), 158-179.

Ramalingam, V., and S. Wiedenbeck (1998). Development and validation of scores on a computer programming self-efficacy scale and group analyses of novice programmer self-efficacy. *Journal of Educational Computing Research*, *19*(4), 367-381.

Transue, B. M. (2013). Connectivism and information literacy: Moving from learning theory to pedagogical practice. *Public services quarterly*, *9*(3), 185-195.

Van Harsel, M. et al. (2020). Examples, practice problems, or both? Effects on motivation and learning in shorter and longer sequences. *Applied Cognitive Psychology*, *34*(4), 793-812.

Western Governors University. (2021). Connectivism Learning Theory. Available at https://www.wgu.edu/blog/connectivism-learning-theory2105.html#:~:text=Connectivism%20is%20a%20relatively%20new,make%20choices%20about%20our%20learning. Accessed 5 February 2022