Human Factors in Design, Engineering, and Computing, Vol. 159, 2024, 1938–1948
https://doi.org/10.54941/ahfe1005761

A Framework for Developing Collaborative Community Building Tools for Novice Computer Science Students

Daniel M. Olivares, Jakob F. Kubicki, and Katie N. Imhof

School of Engineering & Applied Science, Spokane, WA 99258-0102, USA

ABSTRACT

Students enrolled in introductory computer science courses tend towards individual work because of pedagogical practices discouraging collaboration and a focus on individual assignments. This can discourage new computer science students and may negatively affect persistence in computer science. In contrast, social learning theory research suggests a connection between student success and involvement with the learning community. The research presented in this paper puts forth a framework based on social learning theories and teaching and learning methodologies. Tools using this framework will leverage social computing towards building stronger, more connected, social networks and contribute to greater success in student learning outcomes. Using a learner-centered design method, this framework seeks to maintain a focus on the important design questions: how should learning opportunities be scaffolded in a social computing environment? How can we make people more effective learners and promote peer collaboration? How should we motivate learners to remain engaged and form connections? Based on this, the framework speaks to the following guidelines for successful tools stimulating social interaction in a learning environment: 1) scaffolded activities to structure and stimulate community and tool engagement, 2) motivation generating techniques to increase user interaction, 3) competition and community building features supported through gamification to foster learner success. Tools built from this framework's requirements should promote higher levels of interaction and lead to improved learning outcomes, attitudes, and social connectedness by supporting collaboration and problem solving. Further, the framework developed in this paper sets the foundation for future research testing the efficacy of interventions built around a social computing hub where problem solving takes place.

Keywords: Social computing, Social learning, Gamification, Software framework, Community building interfaces, Computer science education

INTRODUCTION

Motivation

Historically, introductory computer science courses are designed with individual student work in mind and, as a result, this pedagogical method can discourage collaboration by focusing on individual assignments (Hundhausen et al., 2008). For new computer science students, this can

discourage collaborative learning and may negatively affect persistence in computer science (Rosson et al., 2011). This historical approach is at odds with what social learning theory and related research suggest are beneficial for learner success. Specifically, that social interaction and involvement in the learning community can be positively associated with greater student success, e.g., see (Astin, 1999; Kolb and Kolb, 2005). Collaboration with peers and mentors can play a key role in increasing social interaction and involvement and help form stronger connections with the learning community. As a result, this could also promote the likelihood of learner success.

We also see from a study analyzing student-used collaboration tools (Ying and Boyer, 2020) that current collaboration tools widely utilized by students are not meeting students' current needs. This highlights another barrier to collaboration and community building in novice computer science student learning environments. A result of this lack of a widely utilized collaboration tools geared towards novice learners is that students tasked to collaborate on a project often use external applications (e.g., "Discord," n.d.; "Slack," n.d.) that may not meet their needs to complete tasks and share code – this can hamper the collaborative learning process. Further, many students end up making use of non-coding applications in their attempt to collaboratively code to overcome steep learning curves found in more coder-focused tools (Ying and Boyer, 2020). Insufficient or overly complex tools are not the only barrier to learner collaboration. There are also concerns surrounding informal and unstructured collaboration tools. For example, failing to learn the material properly, developing poor coding practices, or code plagiarizing are of concern and continue to be of interest in ongoing research on code plagiarism detection (Cheers et al., 2020). Despite these barriers and concerns, there is still value in student collaboration: student success. Use of collaboration tools have been shown to positively affect the collaborative processes by increasing the range, speed and information content of communication, automating goal tracking processes and by providing additional avenues for information distribution (Knutas et al., 2014b). Indeed, collaborative programming tools do exist and attempt to leverage the benefits of collaborative coding, e.g., see (Boyer et al., 2008; "Replit," n.d.), but these have failings that limit their scope and application usefulness by being limited to specific languages and platforms and/or being closed-source rather than open-source.

Another challenge with use of collaborative programming tools is user adoption and use of new tools leading us to ask which factors do potential users consider most important when adopting a new tool (Palani et al., 2022) and how can we overcome barriers to adoption? This is especially difficult for tools introducing new strategies and/or used as part of the learning environment (Taylor et al., 2018). Studies have shown that providing instructors and students with tutorials, e.g. through a series of workshops or videos that can be viewed online, can help integrate adoption of new tools into courses (Clarke et al., 2010). That may provide a starting point for introducing new tools to the learning community.

Necessity of a Universal Collaborative Community Building Tool: The Foundation for a Solution

These challenges motivate the necessity for universal collaborative coding tools to bridge the gap between currently available tools and resources. General purpose tools currently available have specific niche focuses, e.g., the Visual Studio Code extension Direct Messages ("Direct Messages," n.d.) integrates direct messaging into the coding environment and Microsoft's Live Share ("Microsoft Live Share," n.d.) adds collaborative coding sessions but both are not necessarily designed primarily for novice computer science students.

With these considerations in mind, the framework put forward in this paper focuses on supporting students in learner-centered academic settings and not general-purpose settings. Tools created via from framework are structured to achieve the goal of promoting community building and social interaction through empirically based gamification methodologies and collaborative problem-solving strategies. Our research aims to answer the following three primary research questions with regards to tools created based on the framework.

RQ1: Will learners with higher levels of interaction with the tool show increased involvement with the learning community (peers, instructors)?

RQ2: Will increased interactions with the tool be associated with an increase in behaviors shown to positively affect student success?

RQ3: Will increased indicators (interaction levels, behaviors) be associated with a positive change in student success in computer science courses?

RELATED WORK

A survey of existing tools illustrates an array of prior works with features that overlap with our framework goals presented in this paper but also have key issues preventing full usefulness in our proposed scenario. Microsoft's Live Share ("Microsoft Live Share," n.d.), for example, enables code sharing and collaboration but has limitations regarding number of participants. Likewise, Cloud IDE Codeanywhere ("Codeanywhere," n.d.) has cloud-based collaborative coding but has constraints like subscription fees and participant limits. Codeshare ("Codeshare," n.d.) is another online collaborative coding tool. Unlike Codeanywhere, Codeshare is free but has other issues, e.g., advertisements in the coding environment and a broad target use cases like coding interview sessions. In contrast, Replit ("Replit," n.d.) is a collaborative coding environment that touches on many of the collaborative aspects fitting in our framework but has its own limitations. It is an online browser-based IDE and lacks the ability to integrate features required as part of our framework, e.g., integrated scaffolding of problem-solving activities or social interaction-based gamification strategies addressing motivation or engagement.

Social interaction-focused applications not originally designed for coding have also been utilized in the classroom. Discord ("Discord," n.d.) is an instant messaging social platform app with studies demonstrating that students found the instant messaging and text channels to be powerful

collaboration tools with 68% of computer science students reporting that Discord helped team members assist each other (Bridson et al., 2022). Slack ("Slack," n.d.) contains similar features such as chat rooms organized by topic, private groups, and direct messaging. However, both Discord and Slack were not built for a classroom environment and lack built-in features designed with computer science students in mind (e.g., coding challenges, anonymous posting, office hours, etc.). Piazza ("Piazza," n.d.) is a widely utilized classroom tool that contains many useful features, but the user interface, formality of its design, and lack of direct messaging limit approachability for novice computer science students (Bridson et al., 2022).

A significant issue with these tools is that they tend towards being overly complex and are not intended for novice programmers or that they are not able to be integrated as part of the learner's problem-solving and social environment efficiently. They may also have usage limits, may be designed more towards niche programming scenarios, or may have a fee structure attached for use. The necessity for a universal collaborative tool dictates combining elements from collaboration tools that have been proven to boost user engagement and promote social collaboration between students. Helpful collaboration features integrated directly into a single environment can allow students to work more easily in groups, pairs, or with instructors on collaborative or social tasks; all things which have been proven to boost overall productivity for students (Boyer et al., 2008).

RESULTS

Elements of a Framework for Developing Collaborative Community Building Coding Tools

For the purposes of this research, a collaborative community building tool is one that meets the needs of novice computer science students in their problem-solving environment. The following framework emphasizes three primary elements necessary to structure learning, stimulate social interaction, and keep students engaged in a learning environment.

Scaffolding. Scaffolded activities are used to structure and stimulate community engagement and to encourage interaction with the tool and the community. This is a key feature that is lacking in related general-purpose tools commonly used by novice computer science students. Many of these tools, while supporting collaborative tasks, are general use tools and do not provide necessary scaffolding geared towards learning. Scaffolded learning opportunities are a key foundation for this framework's success (see scaffolding theory: Wood, 1976). Further, research has shown the importance of scaffolding as part of the learning process (Reiser, 2004) and its inclusion as a vital component of methods like game- and problem-based learning (Sharma and Giannakos, 2023). Therefore, scaffolding techniques should be one of the key pillars of a framework used for creating tools to aid novice computer science students and may yield best results if carefully integrated into the learning process (Sharma and Hannafin, 2007).

Motivation. Motivation generating techniques can be used to increase user interaction with software tools. This framework builds on the premise

that tools created for the purpose of aiding novice programmers should be focused on encouraging and building a social learning environment that supports collaboration and problem solving while also providing support for instructor guidance and structured to support many problems new computer science students face in the classroom. Indeed, social context and influences play a role in learner motivation (Cook and Artino, 2016). Prior research shows that not only is social interaction important as part of the learning process (Carter et al., 2017) but also shoes promise when it is the focus of a problem-solving tool in a learning environment. Further, increased interactions with social programming tools may have positive correlations with more frequent social activity, positive attitudes toward peer learning, more closely coupled social networks, and improved performance in some cases (Olivares et al., 2021). In addition, gamification techniques can also positively affect motivation (Cahyono, 2023).

Engagement. Continued engagement in a learning environment is another challenge as demonstrated by ongoing struggles to improve weak engagement in the Computer Science discipline (Morgan et al., 2018). This can not only be an issue for learning but one of adoption of learning tools. One method for capturing the attention of learners to overcome the challenge of user adoption of a new tool is to include gamification as part of the learning environment (Cahyono, 2023; Marín et al., 2018). Likewise, this approach is also supported by results seen in a 2014 study (Knutas et al., 2014a) showing gamification features used to foster collaboration and community. Therefore, the goal of increased engagement can be tackled with tools supporting focused social interaction and gamification features to foster learner success by encouraging competition and community building.

Successful tools founded on primary elements of this framework should: 1) set learners up with scaffolded learning opportunities, 2) motivate learners to interact with the learning community by promoting social interaction, and 3) adopt and remain engaged with the learning materials. The hypothesized benefits include improved learning outcomes, attitudes, and social connectedness.

Developing a Framework-Based Collaborative Community Building Coding Tool

Ultimately, the framework provides guidelines that can be put into practice as a practical tool such as one developed for a commonly used code editing workspace like Visual Studio Code ("Visual Studio Code," n.d.) as an extension. Visual Studio Code is an ideal environment due to its cross-platform utility (i.e., target users not limited to a single platform) and capabilities to expand and integrate new functionality using extensions. Additionally, its widespread use, adoption, and availability in places like GitHub's Codespaces ("Codespaces," n.d.), e.g., Harvard's CS50 adaptation of this has been used by more than 80,000 users historically (Malan, 2022) emphasize its future value.

Following that motivation, our research follows a mixed methods approach for the design and development of a framework-based tool

to answer RQ1-3. Primarily, research suggesting that the inclusion of scaffolded social interaction (Boyer et al., 2008; Carter et al., 2017; Olivares et al., 2021) and gamification(Knutas et al., 2014a; Marín et al., 2018) features can benefit learner success. The framework supports this research to encourage competition and community building so our tool, the Coding Social Hub (CSH, see **Figure 1**), leverages the benefits of including a social aspect into the problem-solving environment. From that increased social interaction and involvement in the learning environment we ultimately aim towards the goal of improved success among learners. Further, development of the CSH is designed to evaluate gamification approaches designed to increase engagement will include use badges and challenges to encourage student competition, collaboration and towards fostering a stronger learning community. For example, when students complete coding challenges (participation assignments, collaborative tasks, etc.), they would be rewarded with a relevant badge. Leaderboards tie the gamification features to the community building and social interaction aspects of the tool and situate interactions and achievements within the learning community to provide motivational context to learners.

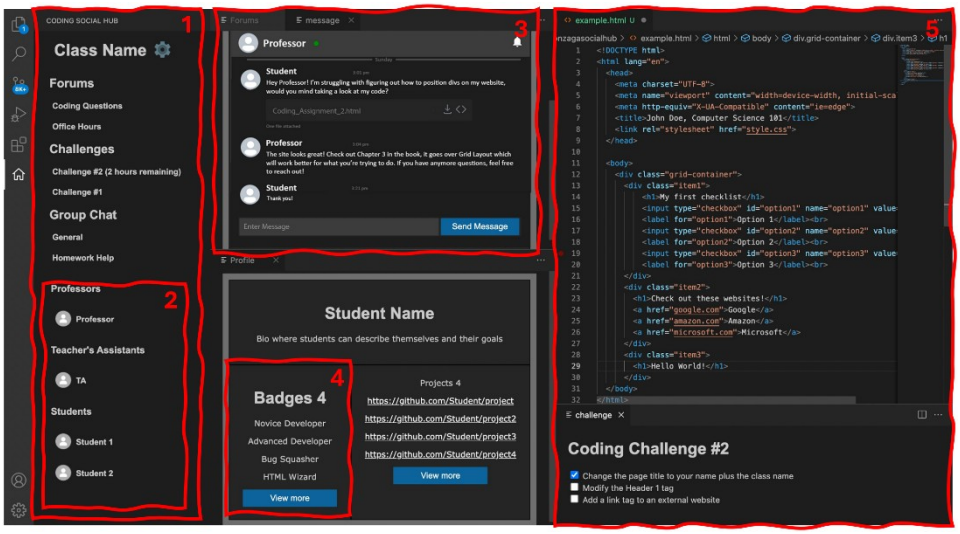

Figure 1: Coding social hub as an extension for visual studio code. The left pane (1) shows forum categories, coding challenges, chat groups, and users (2)[b,c]. The middle pane (3) shows a collection of open chat conversations [b,c] and the current user's profile highlighting a user bio, recent badges (4)[b,c], and projects. The right pane (5) shows an open coding challenge[a,c] and expands on the current coding challenge tasks (5). [Framework elements: [a]*Scaffolding*, [b]*Motivation*, [c]*Engagement*].

METHODS

Implementation and Testing a Framework-Based Collaborative Community Building Coding Tool. The design approach and testing of the tool described in this paper follow an iterative, user- and learner-centered design approach

(Hsi and Soloway, 1998; Norman and Draper, 1986) and the goal of designing with the user and the learner in mind. A challenge the design of this software tool must overcome is not just one of adoption (Palani et al., 2022; Taylor et al., 2018) but also one of keeping the learner at its center because learner success is at the core of our tool's goals. User-centered design is the base of our design approach, but learner-centered design maintains focus on the important questions we must keep in mind: how can we make people more effective learners, adopt our tool (students and instructors) and promote/motivate peer collaboration? How should we scaffold learning opportunities? How should we motivate learners to remain engaged? (Hsi and Soloway, 1998)

Tool Design. The framework-based tool interface prototype (**Figure 1**) uses a variety of libraries available in the Visual Studio Code environment. Initial design iterations are developed as a web-based interface using HTML, CSS, and TypeScript/JavaScript on the front-end and NodeJS with MongoDB on the back end. Evaluation of these designs' ease of use, speed, and resource usage to maintain usability for the user in the form of a lightweight extension is also a part of this research.

Participants. Each prototype design iteration is followed by user testing sessions with 3–5 target users (computer science students, faculty) to gauge initial user feedback and to further refine the tool design. Initial participant pools consist of CS1 (introductory) level students, but the chosen tool platform supports a large array of programming languages and therefore other course topics (e.g., web development). With that in mind, typical participants will be first and second year Computer Science students and expand to include third and fourth year when participant testing expands to additional course topics. Additional participants will be recruited and included in future studies with classroom-wide testing (typically 20–30 students per section).

Iterative Design Process. Early prototype feedback sessions focus on individual tool features to elicit feedback on specific interface elements and usability and will be situated around a scenario to provide context for the individual feature usage in the overall environment, e.g., the entirety of **Figure 1** along with a scenario provided for context to frame the user mindset within the learning environment while eliciting feedback on a specific feature subset.

Data Collection. Data collected during these studies will consist of audio and video recordings, semi-structured interviews, user surveys (pre-mid-post when viable). Additionally, during testing of the live interface, logged interaction data will answer **RQ1-2**. Anonymized collection of student success metrics (grades) will help assess **RQ3**.

Tool Efficacy. A series of larger scale testing will be conducted to validate the efficacy of these tools with deployment in introductory computer science courses. Individual testing of each of the three components of the framework using logged data and survey results will address assessed effectiveness of each component.

CONCLUSION

Following the research motivating the framework, one can conclude that supporting these guidelines should, in theory, result in positively associated results. That is, learners interacting with tools that support scaffolded interactions and include motivation- and engagement-supporting features will be more likely to be involved in the learning community and therefore enjoy greater learning success. This leaves the next steps of future work to test these hypotheses with tools created using the framework guidelines. The goals of our research are to accomplish three key contributions to further educational technologies for computing education

A foundational framework for building collaborative community building coding tools. Prior research has shown evidence of ways in which we can structure and stimulate desired behavior (e.g., scaffolding, gamification) and the potential benefits that can come as a result (e.g., stronger learning community, higher likelihood of success). The framework presented in this paper is a starting point.

Implementation and validation of a collaborative community building tool. Based on research into collaboration strategies and tools, we believe the presented tool is a promising way to promote user collaboration and communication embedded as a vital part of the problem-solving process in computer science classes. The described extension (**Figure 1**) leverages the framework and lessons learned from prior works to improve student and teacher communication, foster a sense of community, and success within the classroom.

Empirical effectiveness of the tool. As a result of the development, user testing, and data collection during a semester-long deployment study, we expect to provide empirically based evidence on the tool's effectiveness as a collaboration and community-building tool embedded in a coding environment and whether it improves community communication and collaboration rather than hinder it.

FUTURE WORK

Future work necessitates deploying a study in a classroom environment over the course of a semester. A benefit of Visual Studio Code as the programming environment is that specific courses and programming languages should not necessarily be a limit because of the wide array of languages supported by Visual Studio Code. This opens the possibility of adopting this tool in both introductory and advanced computer science courses. Likewise, a key next step is to make the tool available to other institutions for a broader range of participants as part of the testing process. Plagiarism is also an ongoing concern in a computer science classroom (Cheers et al., 2020). This issue is further compounded by the recent adoption of Artificial Intelligence tools like ChatGPT and the question of detecting (e.g., Mouli et al., 2024) or using it as a learning tool (Cipriano and Alves, 2023). Interestingly, "guardrailed" approaches using AI tools have already received positive reception (Liu et al., 2024) and warrant research on possible integration into the framework for building collaborative community building tools.

REFERENCES

Astin, A. W., 1999. Student involvement: A developmental theory for higher education. Journal of College Student Development 40, 518–529.

Boyer, K. E., Dwight, A. A., Fondren, R. T., Vouk, M. A., Lester, J. C., 2008. A development environment for distributed synchronous collaborative programming, in: Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE '08. Association for Computing Machinery, New York, NY, USA, pp. 158–162. https://doi.org/10.1145/1384271.1384315

Bridson, K., Atkinson, J., Fleming, S. D., 2022. Delivering Round-the-Clock Help to Software Engineering Students Using Discord: An Experience Report, in: Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1, SIGCSE 2022. Association for Computing Machinery, New York, NY, USA, pp. 759–765. https://doi.org/10.1145/3478431.3499385

Cahyono, D., 2023. Gamification for Education: Using LexiPal to Foster Intrinsic and Extrinsic Learning Motivation of Students with Dyslexia, in: Proceedings of the 14th International Conference on Education Technology and Computers, ICETC '22. Association for Computing Machinery, New York, NY, USA, pp. 32–38. https://doi.org/10.1145/3572549.3572555

Carter, A. S., Hundhausen, C. D., Adesope, O., 2017. Blending Measures of Programming and Social Behavior into Predictive Models of Student Achievement in Early Computing Courses. ACM Transactions on Computing Education 17, 1–20.

Cheers, H., Lin, Y., Smith, S. P., 2020. Detecting Pervasive Source Code Plagiarism through Dynamic Program Behaviours, in: Proceedings of the Twenty-Second Australasian Computing Education Conference. Association for Computing Machinery, New York, NY, USA, pp. 21–30.

Cipriano, B. P., Alves, P., 2023. GPT-3 vs Object Oriented Programming Assignments: An Experience Report, in: Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1, ITiCSE 2023. Association for Computing Machinery, New York, NY, USA, pp. 61–67. https://doi.org/10.1145/3587102.3588814

Clarke, P. J., Allen, A. A., King, T. M., Jones, E. L., Natesan, P., 2010. Using a web-based repository to integrate testing tools into programming courses, in: Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion, OOPSLA '10. Association for Computing Machinery, New York, NY, USA, pp. 193–200. https://doi.org/10.1145/1869542.1869573

Codeanywhere [WWW Document], n.d.. Cloud IDE · Online Code Editor · Codeanywhere. URL https://codeanywhere.com/index

Codeshare [WWW Document], n.d.. Codeshare - Share code in real-time with developers in your browser. URL https://codeshare.io/

Codespaces [WWW Document], n.d.. GitHub Codespaces · GitHub. URL https://github.com/features/codespaces

Cook, D. A., Artino, A. R., 2016. Motivation to learn: an overview of contemporary theories. Med Educ 50, 997–1014. https://doi.org/10.1111/medu.13074

Direct Messages [WWW Document], n.d.. Extension for Visual Studio Code - Direct Messages for VS Code. URL https://marketplace.visualstudio.com/items?itemName=techsyndicate.vscode-dms

Discord [WWW Document], n.d.. Discord - Group Chat That's All Fun & Games. URL https://discord.com/

Hsi, S., Soloway, E., 1998. Learner-centered design: specifically addressing the needs of learners. SIGCHI Bull. 30, 53–55. https://doi.org/10.1145/310307.310374

Hundhausen, C. D., Narayanan, N. H., Crosby, M. E., 2008. Exploring studio-based instructional models for computing education, in: Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education, SIGCSE '08. Association for Computing Machinery, New York, NY, USA, pp. 392–396. https://doi.org/10.1145/1352135.1352271

Knutas, A., Ikonen, J., Nikula, U., Porras, J., 2014a. Increasing collaborative communications in a programming course with gamification: a case study, in: Proceedings of the 15th International Conference on Computer Systems and Technologies, CompSysTech '14. Association for Computing Machinery, New York, NY, USA, pp. 370–377. https://doi.org/10.1145/2659532.2659620

Knutas, A., Ikonen, J., Ripamonti, L., Maggiorini, D., Porras, J., 2014b. A study of collaborative tool use in collaborative learning processes, in: Proceedings of the 14th Koli Calling International Conference on Computing Education Research, Koli Calling '14. Association for Computing Machinery, New York, NY, USA, pp. 175–176. https://doi.org/10.1145/2674683.2674706

Kolb, A. Y., Kolb, D. A., 2005. Learning Styles and Learning Spaces: Enhancing Experiential Learning in Higher Education. Academy of Management Learning & Education 4, 193–212.

Liu, R., Zenke, C., Liu, C., Holmes, A., Thornton, P., Malan, D. J., 2024. Teaching CS50 with AI: Leveraging Generative Artificial Intelligence in Computer Science Education, in: Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 2, SIGCSE 2024. Association for Computing Machinery, New York, NY, USA, p. 1927. https://doi.org/10.1145/3626253.3635427

Malan, D. J., 2022. Standardizing Students' Programming Environments with Docker Containers: Using Visual Studio Code in the Cloud with GitHub Codespaces, in: Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 2, ITiCSE '22. Association for Computing Machinery, New York, NY, USA, pp. 599–600. https://doi.org/10.1145/3502717.3532164

Marín, B., Frez, J., Cruz-Lemus, J., Genero, M., 2018. An Empirical Investigation on the Benefits of Gamification in Programming Courses. ACM Trans. Comput. Educ. 19, 4:1–4:22. https://doi.org/10.1145/3231709

Microsoft Live Share [WWW Document], n.d.. Use Microsoft Live Share to collaborate with Visual Studio Code. URL https://code.visualstudio.com/learn/collaboration/live-share

Morgan, M., Butler, M., Thota, N., Sinclair, J., 2018. How CS academics view student engagement, in: Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2018. Association for Computing Machinery, New York, NY, USA, pp. 284–289. https://doi.org/10.1145/3197091.3197092

Mouli, C., Kotteti, M., Lal, R., Chetti, P., 2024. Coding Integrity Unveiled: Exploring the Pros and Cons of Detecting Plagiarism in Programming Assignments Using Copyleaks. J. Comput. Sci. Coll. 39, 61–69.

Norman, D. A., Draper, S. W., 1986. User Centered System Design: New perspectives on human-computer interaction. Lawrence Erlbaum, Hillsdale, NJ.

Olivares, D., Hundhausen, C., Ray, N., 2021. Designing IDE Interventions to Promote Social Interaction and Improved Programming Outcomes in Early Computing Courses. ACM Transactions on Computing Education (TOCE). https://doi.org/10.1145/3453165

Palani, S., Ledo, D., Fitzmaurice, G., Anderson, F., 2022. "I don't want to feel like I'm working in a 1960s factory": The Practitioner Perspective on Creativity Support Tool Adoption, in: ACM Conferences. Presented at the CHI '22: Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, pp. 1–18.

Piazza [WWW Document], n.d. Piazza • Ask. Answer. Explore. Whenever. URL https://www.piazza.com

Reiser, B. J., 2004. Scaffolding Complex Learning: The Mechanisms of Structuring and Problematizing Student Work. Journal of the Learning Sciences 13, 273–304. https://doi.org/10.1207/s15327809jls1303_2

Replit [WWW Document], n.d.. Replit - Build software faster. URL https://replit.com/

Rosson, M. B., Carroll, J. M., Sinha, H., 2011. Orientation of Undergraduates Toward Careers in the Computer and Information Sciences: Gender, Self-Efficacy and Social Support. ACM Trans. Comput. Educ. 11, 14:1–14:23. https://doi.org/10.1145/2037276.2037278

Sharma, K., Giannakos, M., 2023. Carry-Forward Effect: Early scaffolding learning processes, in: Proceedings of the 2023 Symposium on Learning, Design and Technology, LDT '23. Association for Computing Machinery, New York, NY, USA, pp. 43–52. https://doi.org/10.1145/3594781.3594786

Sharma, P., Hannafin, M. J., 2007. Scaffolding in technology-enhanced learning environments. Interactive Learning Environments 15, 27–46. https://doi.org/10.1080/10494820600996972

Slack [WWW Document], n.d.. Slack is your productivity platform | Slack. URL https://slack.com/

Taylor, C., Spacco, J., Bunde, D. P., Butler, Z., Bort, H., Hovey, C. L., Maiorana, F., Zeume, T., 2018. Propagating the adoption of CS educational innovations, in: Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2018 Companion. Association for Computing Machinery, New York, NY, USA, pp. 217–235. https://doi.org/10.1145/3293881.3295785

Visual Studio Code [WWW Document], n.d.. Code editing. Redefined. URL https://code.visualstudio.com/

Wood, D. J., 1976. The role of tutoring in problem solving. Journal of Child Psychiatry and Psychology 17, 89–100.

Ying, K. M., Boyer, K. E., 2020. Understanding Students' Needs for Better Collaborative Coding Tools, in: Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, CHI EA '20. Association for Computing Machinery, New York, NY, USA, pp. 1–8. https://doi.org/10.1145/3334480.3383068