Project Proposal

Madeline Burton
mburton40@gatech.edu

***Abstract*—**Two-way feedback is a pedagogical best practice in primary and secondary classrooms. This paper acknowledges existing feedback platforms and argues there is room for a new platform more specifically tailored towards two-way feedback in computer science classrooms, which when used in combination with respectful inquiry will increase student success in these classes. It then outlines a trajectory for the creation of this platform.

1 INTRODUCTION

While it was once the role of a teacher in a classroom to stand and deliver knowledge to pupils, over the past few decades this role has transitioned. Now, the teacher is a mentor who creates opportunities for learners to use their existing knowledge to make predictions and connections in order to better understand new topics (Dochy and McDowell, 1997). With this transition, formative feedback has become an essential tool, serving as an important structure to educate, evaluate, and embolden student learners. Formative feedback – feedback given throughout the learning process to help guide the learner's development – serves to educate the learner by engaging them in their zone of proximal development through respectful inquiry that encourages them to think deeper about a particular subject (Mirzaee and Hasrati, 2014). It is often evaluative, giving a learner a clear understanding of their current place in the development process and sharing specific steps to move forward (Cauley and McMillan, 2010). It is also an essential tool for supporting students emotional engagement and and helping them build self-efficacy through thoughtful, specific, or motivational feedback (Tan et al., 2019; Cauley and McMillan, 2010).

In the past decade, the process of giving feedback has transitioned from paper and ink to digital means. Through the features afforded by word processing software and screen annotation, instructors are able to give more frequent specific feedback that is both convenient and easy to interact with. Learners respond to this feedback more proactively, choosing more frequently to revise their work or engage in dialogue with their teacher to discuss further meaning (Chong, 2019). Through increased dialogue, e-feedback systems are creating more opportunities for students to participate in active-learning. Not only does this two-way feedback increase success in learning outcomes, it "elicits perceptions and judgements, and assists learners to define future actions... ensure understanding, and become independent learners" (Tan et al., 2019). The increase of e-feedback is thus facilitating the role of the modern day instructor.

This process is not unique to education, and is frequently used to support the development of software through code reviews. Software engineers give feedback on each other's code in order to make the product more efficient, reduce confusion in the code base, and gain insight into the work of others (Anderegg, 2020). In both contexts, the value of feedback is evident and whenever possible should be pursued to further one's cognitive and metacognitive growth.

Deep student learning and development is best fostered by ongoing dialogue about learner created artifacts. When formative feedback is designed to support active learning, metacognitive processes can promote deeper and more efficient learning (Price et al., 2010). While it is possible for an instructor to write directed formative feedback with this purpose, the learner's development is enhanced when the feedback is treated as an opportunity for dialogue in which the instructor can leverage techniques of respectful inquiry to strengthen this process even more (Tan et al., 2019). There exist user friendly tools to provide ongoing dialogue on learners' textual works, and industry tools to provide ongoing dialogue about source controlled code, however no platform exists to facilitate this feedback process in an programming context without the overhead cognitive load of mastering a source control system.

There is hole that exists in education technology in the context of computer programming; while tools exist to give one-way feedback at the end of an assignment or to give automated feedback based on predetermined tests or programming best practices, there are no such tools to encourage dialogue and the deeper level thinking and iteration that it produces. Without such a tool the learning process moves more slowly, CS courses may have higher attrition, and learners are not prepared with the higher level thinking skills necessary to solve open ended problems. In the absence of a digital tool to support this dialogue, feedback can still occur. In this instance however, it would only exist in the classroom through oral discussion and thus be limited to the number of learners an instructor could interact with during a class period. Given the prominence of technology in the classroom, it is only logical for this process to be available digitally which would allow dialogue to occur outside classroom hours and support a greater number of students in a timely manner.

2 RELATED WORK

In order for successful two-way feedback to occur, there must exist a tool to facilitate this communication. While there are many tools that facilitate communication between students and teachers, a tool that specifically supports discussion surrounding computer code is lacking from the introductory computer programming classroom.

2.1 Educational Tools

As of 2017, Google Apps for Education (now G Suite) was used by over 70 million students and teachers to support the creation of academic artifacts and dialogue about these products (Fenton, 2017). Google Docs, a product in this suite, is a tool that provides many benefits over traditional paper and ink feedback. Its simple user interface is designed with collaboration in mind. In a study of 65 learners newly introduced to Google Drive, 93.84% felt the tool was easy or very easy to use (Kakoulli Constantinou, 2019). There are numerous features within Google Docs that can be leveraged for dialogue. Most importantly perhaps is the ability to have threaded conversations about a particular portion of a student's written artifact. Task-specific comments such as, "can you think of another example to support this statement?" tied directly to a portion of the work encourage students to think more deeply about their work and are more likely to contribute to extrinsic motivation than either grades or praise (Cauley and McMillan, 2010). Google Doc comments can be instigated by both the instructor and the learner, creating a simple opening for dialogue.

Many Learning Management Systems (LMSs) include the ability to interact directly with a student by leaving comments on the submitted student work. One such system is Canvas's DocViewer embedded in the Speed Grader, which allows teachers to view supported file types and leave annotated comments on top of the student work (Titmus, 2020). Additionally, instructors can leave point annotations – comments relevant to a particular portion of the project – that can be accessed as threaded discussions in a similar fashion to Google Docs. Unfortunately, in Canvas as in many LMSs the user interface is not particularly intuitive on the student side such that they frequently do not know that any comments on an assignment exist negating any possibility of dialogue (Stuhlsatz, 2020; Colburn, 2020).

2.2 Industry Tools

GitHub, Crucible, and BitBucket are all similar platforms created for the specific purpose of hosting developers' code, which excel at managing complex code bases. They do this through a suite of features including version control, change tracking, issue tracking, and collaborative development. When leveraged in education, the issue tracking features of GitHub have been used in a number of studies for peer review and general learner feedback (Feliciano et al., 2016). This allows reviewers to tag a specific section of code and leave comments about this topic in a threaded format. When participating in these studies however students note that there is a significant learning curve one must overcome in order to engage with content through GitHub (Zagalsky et al., 2015; Feliciano et al., 2016; Larsén and Glassey, 2019). These same students shared that if they ran in to issues with GitHub they could more easily solve them by downloading the repository from scratch than trying to resolve the problems in the system. This could be one reason that Glassy (2006) found version control platforms do not encourage students to iterate on designs; in fact students are more likely to procrastinate working on the assignments. These limitations are not unique to GitHub, but applies more broadly to industry source control platforms which are generally over-engineered for educational settings (Anderegg, 2020). The steep learning curve dramatically reduces these tools' effectiveness in an introductory classroom.

2.3 Synthesis

For classes that require general writing assignments, there are many tools available; Google Docs, Microsoft Word 360, and speed-graders built into LMSs are designed to allow instructors to comment, annotate, and make in-line suggestions to student work. They also have been strategically designed to simplify student engagement with instructor feedback. In programming courses, students must submit code with specific formatting and highlighting, code which should be processed by a computer. All of these requirements make the aforementioned tools undesirable solutions.

There are other tools such as GitHub, Crucible, or BitBucket used in the Software Engineering industry to support management of and collaboration surrounding programming projects. While these tools facilitate programming projects, they are not built with the goals of education in mind.

3 PROPOSED WORK

This paper argues for the creation of a new platform in which the benefits of both tools are intersected in order to facilitate two-way feedback about programming artifacts in an introductory computer science classroom. For such a platform to be relevant, it must leverage the values of both educational and software engineering tools. Students in an introductory course are often overwhelmed by the new concepts addressed in the course. For this reason, the tool should be user friendly, taking cues from Google Docs in order to easily stimulate dialogue about students' code that leads to deeper learning opportunities without requiring them to learn a new system. Additionally, such a tool must learn from the success of GitHub and other tools used in industry. The student derived artifacts will be code and thus both the learner and instructor should have the tools to interact with it as such.

Instructors interacting with this tool will be able to easily and efficiently interact with the tool such that the majority of their effort is focused on giving relevant inquiry based feedback to their students. Students will find the tool should be equally accessible in order to encourage two-way feedback interactions as described by Tan et al. (2019). Key features of this web-based tool include:

- syntax highlighting
- line-specific commenting
- the ability to 'reply' to a particular comment as a thread
- comment editing will support markdown
- difference highlighting between submissions

4 DELIVERABLES

4.1 Milestone 1

Milestone 1 will include wireframes and a prototype in which a user may log on, upload a file, and display it with syntax highlighting. Beyond file upload, there will not be additional interactive features. Reviewers will have the opportunity to watch a video walk-through of the project design and log in and interact with the tool as desired.

4.2 Milestone 2

Milestone 2 will attempt to deliver a platform to which a user has the ability to upload a file and may engage with the Interactive Dialogue Features (comments, threaded replies, highlighting, and markdown). Reviewers will have the opportunity to watch a video walk-through of these features and may log on and interact with the tool as desired.

4.3 Final Product

The final project deliverable will include an interactive web platform through which learners and instructors are able to discuss programming artifacts created by the student. Reviewers will be able to interact with all features described in 3. The final project will also include a video demonstration of the features of the platform and a final project paper.

5 TASK LIST

A list of anticipated tasks is available in Table 1. Note that hours are front loaded on earlier weeks. This is to acknowledge external commitments and to allow for the possibility of additional hours in later weeks if necessary.

Table 1—A proposed high level task list of the tasks to complete for the design of Code Dialogue.

Week	Task Number	Description	Estimated Time
6	1	Prepare Web Server	8
	1.1	› Create Repository	1
	1.2	› Create and Connect Heruko Server	3
	1.3	› Create and Connect Database	4
6	2	Develop Data Infrastructure	8
	2.1	› Create File Datastructure	2
	2.2	› Create Comment Datastructure	2
	2.3	› Create User Datastructure	2
	2.4	› Connect User Ownership	2
6	3	Integrate with Google OAuth	4
6	4	Weekly Status Check	0.25
7	5	UI Design and Storyboard	4
7	6	Upload Files	6
7	7	Display File:	15
	7.1	› Develop UI	6
	7.2	› Display File	4
	7.3	› Add Syntax Highlighting	5
7	8	Weekly Status Check & Milestone 1	2.25
8	9	Interactive Dialogue Features	19.5
	9.1	› Add Comments	8
	9.2	› Add Threaded Replies	4
	9.3	› Add Highlighting	4.5
	9.4	› Add Markdown	3
8	10	Weekly Status Check	0.25
9	11	Develop User Dashboard	6
9	12	Develop Diff-ing Engine	12
	12.1	› Add Versioning to Database	2
	12.2	› Compare Versions	10
9	10	Weekly Status Check & Milestone 2	2.25
10	13	Reload and Display Differences	4
10	14	Add Download Feature	3
10	15	Share Files	2
11	16	Final Presentation & Paper	10
		Total Hours:	106.5

6 REFERENCES

[1] Anderegg, C. (2020). The value of code reviews in industry [Private Interview].

[2] Cauley, K. M., & McMillan, J. H. (2010). Formative assessment techniques to support student motivation and achievement. *The Clearing House: A Journal of Educational Strategies, Issues and Ideas*, *83*(1), 1–6. https://doi.org/https://doi.org/10.1080/00098650903267784

[3] Chong, S. W. (2019). College students' perception of e-feedback: A grounded theory perspective. *Assessment & Evaluation in Higher Education*, *44*(7), 1090–1105. https://doi.org/https://doi.org/10.1080/02602938.2019.1572067

[4] Colburn, B. (2020). Feedback empathy research [Private Interview].

[5] Dochy, F. J., & McDowell, L. (1997). Introduction: Assessment as a tool for learning. *Studies in educational evaluation*, *23*(4), 279–98. https://doi.org/https://doi.org/10.1016/S0191-491X(97)86211-6

[6] Feliciano, J., Storey, M.-A., & Zagalsky, A. (2016). Student Experiences Using GitHub in Software Engineering Courses: A Case Study. https://ieeexplore.ieee.org/abstract/document/7883328

[7] Fenton, W. (2017). Google Classroom Could Bridge a Gap in Online Learning. https://www.pcmag.com/opinions/google-classroom-could-bridge-a-gap-in-online-learning

[8] Glassy, L. (2006). Using version control to observe student software development processes. *Journal of Computing Sciences in Colleges*, *21*(3), 99–106. https://dl.acm.org/doi/10.5555/1089182.1089195

[9] Kakoulli Constantinou, E. (2019). *Revisiting the cloud: Reintegrating the g suite for education in english for specific purposes teaching*. ERIC. https://pdfs.semanticscholar.org/f7c7/c1f5c2f1a1e581977b4641f21add1c19dcba.pdf

[10] Larsén, S., & Glassey, R. (2019). RepoBee: Developing Tool Support for Courses using Git/GitHub, In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, Aberdeen, Scotland Uk, Association for Computing Machinery. https://doi.org/10.1145/3304221.3319784

[11] Mirzaee, A., & Hasrati, M. (2014). The role of written formative feedback in inducing non-formal learning among masters students. *Teaching in Higher Education*, *19*(5), 555–564. https://doi.org/10.1080/13562517.2014.880683

[12] Price, M., Handley, K., Millar, J., & O'Donovan, B. (2010). Feedback : All that effort, but what is the effect? *Assessment & Evaluation in Higher Education*, *35*(3), 277–289. https://doi.org/https://doi.org/10.1080/02602930903541007

[13] Stuhlsatz, E. (2020). Feedback empathy research [Private Interview].

[14] Tan, F. D., Whipp, P. R., Gagné, M., & Van Quaquebeke, N. (2019). Students' perception of teachers' two-way feedback interactions that impact learning. *Social Psychology of Education*, *22*(1), 169–187. https://doi.org/https://doi.org/10.1007/s11218-018-9473-7

[15] Titmus, C. (2020). How do i add annotated comments in student submissions using docviewer in speedgrader? https://community.canvaslms.com/docs/DOC-26515-how-do-i-add-annotated-comments-in-student-submissions-using-docviewer-in-speedgrader

[16] Zagalsky, A., Feliciano, J., Storey, M.-A., Zhao, Y., & Wang, W. (2015). The Emergence of GitHub as a Collaborative Platform for Education, In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, Vancouver, BC, Canada, Association for Computing Machinery. https://doi.org/10.1145/2675133.2675284