Developing a Responsive Computer Science Program with a Collaborative Design Model

Dr. Nicole Anderson, Weber State University

Dr. Nicole Anderson is an Associate Professor of Computer Science at Weber State University.

Dr. Linda DuHadway, WSU

Linda DuHadway has been in higher education for many years. She has degrees from Utah State University and received a PhD from the University of Utah with a focus in Computer Science Education. She is actively engaged in bringing a variety of innovative t

Dr. Kyle D Feuz, Weber State University

Kyle Feuz is a Professor at Weber State University in the School of Computing. He earned his Ph.D from Washington State University under the guidance of Dr. Diane Cook in 2014. He also received his B.S and M.S in Computer Science from Utah State University.

Dr. Richard Fry Fry, Weber State University

In 2010, Dr. Richard Fry became the first Computer Science faculty within his University to introduce Community Engaged Learning (CEL) initiatives into the program's curriculum. Ten years later, he continues to partner with both local and global communit

Julie Christensen, Weber State University

Developing a Responsive Computer Science Program with a Collaborative Design Model

We have pioneered a new program designed to provide students with a more flexible timeline for coursework than traditional Computer Science programs offer. From the ground up, CS Flex has incorporated collaborative program design and curriculum development in new and innovative ways. In this paper, we share details of the innovative collaboration model that has been and continues to be used to develop the responsive CS Flex curriculum.

Introduction

CS Flex is a mastery-based, flexible paced modality for offering classes in the Computer Science program at Weber State. The set of courses for all students is the same, what differs with CS Flex courses is that they have additional flexibility and accessibility compared to traditional offerings. In our program, Computer Science (CS) students can use a combination of CS Flex, traditional online, and face-to-face courses to meet CS degree requirements.

The CS Flex modality is an asynchronous online format. Materials are available to students when it is convenient for them and they are able to move through the courses at a flexible pace. Students are able to accelerate or decelerate with respect to the default course pace depending on their learning needs and life schedule. Being mastery-based means that students progress to new content and learning only when they have a firm understanding of the prior topics.

In other work, we have collected and presented data on individual student outcomes and many of the lessons learned on supporting students from our first five years using the CS Flex model [1]. When we assess the success of the CS Flex program, we believe we need to assess not only student success in terms of content knowledge, but also in terms of other important skills including collaboration. While one might assume a variable paced, asynchronous online learning modality would offer no opportunities for collaboration, we have strategically incorporated collaboration throughout this program, including team collaboration in program design and faculty-to-faculty collaboration in curriculum development.

For the remainder of this paper, our focus is not detailing the CS program as a whole or its success in terms of student technical outcomes, but rather to share some of the collaboration methods we have employed with positive outcomes in the development and delivery of the CS flex courses. Due to the nature of this program, we needed to think deeply about how to incorporate collaboration throughout the process for both students and faculty. Here we share some of the collaboration strategies we used as we believe they are useful to incorporate not only into the continued development of our CS Flex curriculum, but can be replicated and applied to any course or program development process. We believe that each of these collaboration

methods is useful in its own right. They may be used independently or adopted together as each instructor or team sees fit.

Background and Related Work

It is relatively easy to find discussion of instructor-assigned student collaboration, however faculty-faculty collaboration around teaching is less frequent, at least in academic literature. Prior studies indicate that collaborative learning in Computer Science courses is often restricted to loosely structured student group work, with assessment focusing on the technical outcomes. [2]. Even though literature frequently suggests that collaboration is important and useful, and that it in fact increases students' learning outcome and performance [3] when collaborative techniques are employed, they are often not well guided in student work. One contributing factor may be that instructors themselves lack the pedagogical know-how to effectively design and guide collaborative and cooperative learning work [4].

Many of the faculty that are part of the CS Flex design team have experimented and had areas where they have found success with integrating collaborative learning in their courses. For example, Feuz et al. provided experiences for peer learning through required exposure to other student's code via discussion boards [5]. Anderson [5] integrated the use of Pair^2 learning methods involving paired instruction and paired programming. These past experiences, among many others shared by the CS Flex development team, have helped guide discussions of collaboration methods in CS Flex. The team believes that collaboration is a key to good design, not just a preference.

Collaboration in Course Development

In CS Flex, collaboration is a core component that begins with instructors. All course development in this modality is done in pairs. This is reminiscent of the pair programming paradigm, with a driver and a navigator of different modules of each course. Like in pair programming, there are two sets of eyes on each aspect of what is produced. One developer is assigned as lead developer for each module with the other instructor giving feedback and refining what is produced. It diverges from that paradigm in that development is often asynchronous, but both developers impact all aspects of each course through development or review. A more traditional code review may provide another way of thinking about the development process. This structure applies to both initial course development and ongoing refinement or overhaul of courses as needed.

In the CS field in general, one aspect that is constant is change. Technologies and curriculum evolve and our courses must evolve to fit the changing needs. The CS Flex program is often on the leading edge of integration of curriculum changes. For example, in our department these

courses were first to integrate new course assessment practices and new course syllabi requirements defining student learning objectives. This timely integration can likely be attributed to the collaborative nature of the program which provides additional accountability that comes with that collaboration coupled with the funding provided by the program to quickly integrate changes and keep courses current.

Courses are reviewed frequently and refined as needed. For example, bi-weekly meetings between developers and instructors are undertaken during the first course offering to address any necessary refinements. In addition, every three years a formal review and refresh of each course is undertaken to make sure a course continues to meet the latest standards, best practices, and technical needs for the given course. Even if there have been no major changes to standards, best practices or technical needs a course refresh should update approximately 50% of the content. When faculty have finished making changes to a course, the CS Flex coordinator also reviews the content to make sure agreed upon standards are met, and faculty are given monetary compensation for their work on course development and updates.

Collaboration in Course Delivery

Collaboration goes beyond course content creation. At delivery time, the course instructor may or may not be one of the course designers. This provides a new set of eyes reviewing course content and flow. The course instructor is encouraged to provide feedback to the course designers anytime the course is taught by placing the feedback directly into the course template. As described previously, the first and third times a course is taught after being created or updated the instructor and developers meet on a regular basis to discuss any issues or potential improvements. Another benefit of this collaboration is that it can assist new faculty in understanding the norms of the program as a whole and the specifics of the course content for a class that may be new to them. The number of hands involved in course evolution grows over time as new instructors teach the course.

Even for seasoned instructors, some prefer to teach a CS flex version of new courses, especially for courses they are coming up to speed with. This allows them to see and work through course content created by two experts in the area. This can also make having multiple preps in a given term more manageable, as an instructor that is new to a course isn't required to create all course content from scratch, but instead receives well-reviewed, thoughtfully crafted course content.

Of course, not all instructors are happy to use materials created by others. For this reason, some instructors in our department prefer not to teach CS flex courses and opt out of this modality. But for others, this is a huge benefit.

Instructor-Guided Student Collaboration

Part of the materials that have been created for the CS Flex program include documentation on methods that can be utilized for instructor-guided student collaboration. The method below has been created and employed in our CS Flex software engineering courses. Since the CS Flex courses are variable paced, one may see this reflected in the approach taken to creating teams for group work. When a CS Flex course developer decides to integrate this type of collaboration, the information below is included in the instructor notes for the module where team work begins so that any instructor can successfully implement this strategy.

Instructor Notes for Start of Instructor-Guided Student Collaboration:

> It is time to form teams. The rest of the work this semester will be done by teams. Teams will be 3-4 people. You will work together to complete Sprints 1-3.
>
> Because students move through this class at different speeds, there may be some delay in forming a group. We will pause here until there are enough students to form a team. Track shifts will be added to keep the due dates current.
>
> Before submitting this assignment, be sure to complete the Scrum Training Series outlined in the other assignments in this module.
>
> All you need to do for this assignment is to complete the other assignments in this module and let me know you are ready to work on a team. Please enter "Ready" in the text entry box and submit.
>
> Challenge Activity: Form Teams: This activity works as a stopping point for students. The only thing required in this assignment is for students to say they are ready. Leave this assignment ungraded until the student is placed on a team. Once there are three or four students, form a team.
>
> Some students will have to wait to be placed on a team. If this assignment becomes past due while they wait, do a track shift to keep the student current. Use the Place Hold track shift for this purpose. It will shift the due dates but not count towards the student's optional or total track shifts.
>
> Forming Teams: There are several steps required when a team is formed.
>
> *The following instructions are based on using Discord for team communications, Trello for project management, and Google Drive for storing team documents. If you decide to use different tools for any of these, you will need to change some of the messages sent to students listed below. Also see the notes listed at the bottom of this page for other updates that will be needed.

1. Make sure they completed the Scrum Training Series. A score of 80% is required for completion.
2. Create a private voice channel in Discord for students to have team meetings
    - Add each member of the team to this channel
3. Create a private text channel in Discord for students to have team chat
    - Add each member of the team to this channel
    - Post a welcome message. For example:

    Hi Team 3. This is a text channel for you to communicate as a team. It is private. Only your team members and I are able to see this channel. There is also a discord room for you to meet in, Team 3 Room.
    Team 3 Members:
    @Abe Johnson
    @Sally Walton
    @Marvin Waterton
4. Set up a team folder on Google Drive for students to store team documents
    - Share the folder with each member of the team
    - Make sure each team member has edit access to the folder
    - Send a message when you share the folder. For example:

    This is your team folder for CS [course number]. Each member of the team has edit access to the folder. Use this folder for developing and storing your analysis and design documents for the Project Tracking System.
5. Set up a Trello board for students to track tasks
    - Set the board to private
    - Add each team member to the board
6. Create a group in Canvas. There are group assignments in Modules 6-9 that will only work if a group with team members has been set up.
7. Shift tracks as needed to get all members on the same track.
8. Grade the Challenge Activity: Form Teams for assigned members.
    - Add an informative comment. For example:

    Team 3 has been created!
    Abe Johnson
    Sally Walton
    Marvin Waterton
    You can now begin your work as a team. The first thing to do is find a time for your Sprint 1 Planning meeting. Having this conversation as outlined in the Scheduling the Sprint 1 Planning Meeting assignment is due this Thursday. You can have this conversation and future team conversations in the discord channel that has been set up for Team 3.

> Your current track matches the rest of the team. You will all have the same due dates.
> Shifting tracks during the rest of the semester will likely require being assigned to a different team.
> I will be setting up a team board on Trello, a shared folder in google drive, and team discord text and voice channels. Invitations to the Trello board and the shared folder will be sent to your WSU email.
> Please let me know if you have any questions.

> There are times when a team of two is necessary. In that case, less work is required. See Team of Two Notes for more information.

Significant work has been put into guiding new instructors through this method of guided student collaboration. These guidelines have been adjusted over time as experiences have been shared by multiple CS Flex instructors to provide the best experience possible for both faculty and students.

Program Level Collaboration

With a broader lens, we see that collaboration also happens at the program level. We have, as a team, developed a template for all courses. This involves common methods of navigating each course in the learning management system, as well as common policies, such as how often a student can shift the timeline of a course using acceleration or deceleration to complete the modules. We include an instructional designer as part of this development process to ensure we use best practices in the design. As a team, we make decisions on key aspects, including the components required in each course and the thresholds for mastery-level performance. These decisions are data-driven and dynamic; we adapt policies as we assess their effects.

A key benefit of this alignment in course design is that students are able to focus on learning the material specific to each course rather than focusing, especially initially, on the course setup, navigation, and policies. They know what will happen in every course if they turn in an assignment late. They know how to take the exams. They know where to find the syllabus. This doesn't mean there is no flexibility in course design. For example, some courses are project-based and others use exams as their primary assessment mechanism, some require group work and others only individual work. But the general flow and policies of the course are consistent across the program.

A template is provided for each CS Flex Canvas course. Each module that is part of a CS Flex course follows a similar progression. Below is an example of a module introducing Class Diagrams in an introductory Software Engineering course.

Figure 1: Sample Module from CS Flex Canvas Course

Several common elements are shown in Figure 1. Each module starts with instructor notes (an example of instructor notes is shared later in this paper). The first student visible element is a module overview which outlines the module and shares specific student learning outcomes. Next we have one or more Learning Activities, Try It Out activities, and a Challenge Activity. The Learning Activity provides materials for understanding new topics and beginning practicing new skills. Try It Out activities reinforce and allow more in-depth application of the skills, and finally Challenge Activities involve integration of the new skill into their larger toolset and allow them to put it into practice producing a meaningful artifact. This progression is followed in each and every course and has been developed by our team over time after sharing experiences through many iterations of our courses. Since, as you see here, entrance to a module is based on completion and mastery of a prior module, we also provide a look ahead section so students are not stuck in their progress waiting for an instructor to grade a prior module before they can move on to the next module. This is a perfect example of a lesson learned through trial and error - we first learned that quick feedback was needed to allow progression and then developed this method to allow students to progress an appropriate amount while waiting for feedback on their prior module. We came together, discussing not only the challenges faced in allowing students to move at their own pace but also various ideas for resolving the problem. Our team decided on new methods to implement by majority vote after significant debate and discussion. We often circle back to past decisions to discuss how they are working and whether modification is

needed, always with the goal of being responsive in our design process. This allows us to have group buy-in on the decisions made that impact our course design.

Looking at the CS Flex course template as a whole, we can see that it lays out common items to be included in each CS Flex course including standard information such as the course description, learning objectives and textbook information. It also lays out the CS Flex model, with a section describing each aspect of the CS flex model.

- Meaningful student/teacher interaction
- Flexible scheduling that promotes acceleration
- Progress based on mastery of material
- Module delivery of course material

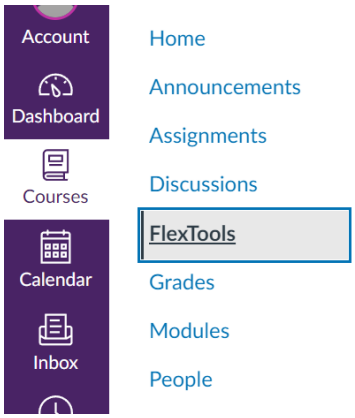
The section on Shifting Tracks, a method we have collaboratively designed to allow students to speed up or slow down a course, dives deeply into that process and gives information about the tools students can use in any CS flex course. Specifically, track shifting relies on FlexTools, which are an add-on to our LMS (Canvas) that have been developed by a subset of our CS Flex team. Let's take a look at this section of our syllabus template to demonstrate the level of collaborative thought put into each portion of the syllabus template.

While we don't expect the details of the track shift process to be replicated in other programs, seeing this piece of the syllabus may help you understand the depth of collaboration that has gone into this development as the details here are an example of the level of detail we decide upon as a team.

Shifting Tracks

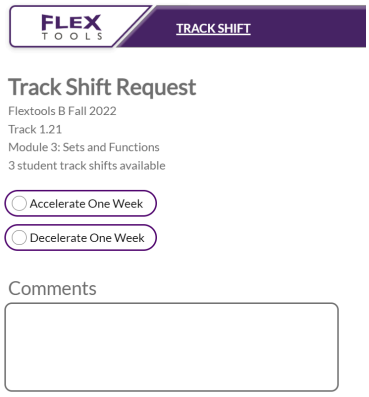
You can shift tracks. There is no limit to how many times you can shift to earlier tracks (accelerate). You have three optional track shifts that can be used for deceleration. There are also track shifts available when needed to achieve mastery. However, you cannot shift more than eight tracks past your initial track (decelerate).

To request a track shift, click on "FlexTools" in the Canvas left menu.



Size: 219 x 240 Alt txt: FlexTools link in Canvas course menu

Once you enter FlexTools you will be presented with the option to request a track shift. It will look similar to what you see here.



Size: 334 x 378 Alt txt: Example track shift request page

After your track shift has been approved or denied, you will receive a Canvas message. Contact your instructor with any questions about the track shift.

Shifting to get ahead

You can shift to earlier tracks to accelerate your progress. Anytime you complete a module or exam one week early, you can request a track change. This will shift all the remaining due dates and can help keep you on pace for early completion. It also adds one additional optional track shift that can be used later. For example, a student who accelerates one week, would have four optional track shifts.

Shifting to spend more time

You have three optional track shifts that can be used throughout the semester. These can be used to shift to a later track. This will shift all the remaining due dates by one week, which will give you more time to work on the current module. To use an optional track shift, you must submit a request at least 24 hours before the module or exam due date. These optional track shifts count toward the maximum of eight.

Shifting to take a break

You can shift to a later track to take a break from the course. This is called a pause. You can have one pause per semester. The pause can be for 1-3 weeks. This will shift all the remaining due dates. This provides a way to pause the course to focus on other things. If you would like to take a break, send an email to csflex@weber.edu and the instructor. Include information about when you want to start the pause, what you will be focusing on instead of this class, and how long of a break you will be taking (1-3 weeks). Each week you pause the course counts towards the maximum of eight.

Module 1 must be successfully completed before taking a pause. If you need time before starting the course, you can register for a section that has different start dates.

Shifting for mastery

If, at some point, you do not reach mastery on a challenge activity or exam, a track adjustment may be made as long as you have not exceeded the maximum eight track shifts. Please see the section on mastery of material below.

Non-submission

There are no track shifts available for non-submission. It is expected that you are engaging in the course each week and turning in the work as expected. If work is not submitted within the late period, a score of zero will be assigned for the activity. Contact the instructor right away. You will need to demonstrate mastery of the material in the current module before moving on to the next module.

Collaboration Beyond our Program Team

The team involved in this work extends beyond our department, as we collaborate across campus. The development team includes a dedicated student advisor with a unique skillset for understanding our students' needs and assisting them with joining the cohort, enrolling in courses and following the CS Flex guidelines. It also includes an instructional designer with expertise in the ins and outs of our learning management system and training in online learning. We value this input from those across campus that assists us in understanding students' needs and adopting best practices in online course design. CS Flex's innovative and non-traditional approach to course delivery requires additional support from across campus, including the Registrar's office, Online and Continuing Education, and Academic Affairs. Collaborating broadly with these offices and personnel responsible for the management and oversight of the university is fundamental to the successful implementation of this new approach. While we collaborate beyond our program team on an ongoing basis, getting the program rolling initially required truly significant effort.

Developing something innovative in a university setting requires broad support. CS Flex pushed against many traditional practices. To introduce this innovative approach to education required funding. Ideas needed to be converted into practices and new curriculum needed to be developed. In our case the Associate Dean was supportive and helpful in identifying funding opportunities and brainstorming which offices would be impacted by the project. Before applying for funding, we needed to garner support from people across campus. To implement the type of modality we were thinking of, we would need to be aware of university procedures and likely require some adaptation. To do this, the principal investigator (PI), reached out to a variety of stakeholders to seek funding.

The first meeting was with the Director of Continuing Education and Online Learning, the Executive Director of the Financial Aid Office, and the Registrar. The Associate Dean also attended (it is helpful to have a champion of the proposal participate). The response was 'that is a great idea but it is impossible'. In fact in all of the meetings this was the initial response. And at the word impossible, they were ready to end the conversation. To avoid this, the next step was to ask them if they had any ideas. Maybe something similar could be done, maybe something could be adapted, maybe these experienced administrators could come up with something novel. Then the conversation became interesting as they brainstormed. By the end of the meeting it no longer seemed impossible. Together they were working out ways it might be done. They would end with maybe they could do a little something but we would also need to talk to … and here they would list a few other offices that would be impacted.

The PI then set up a meeting with this new group of people and went through the process again. The meetings were very similar, first the response that this is a great idea, then that it is not possible, then the brainstorming and coming up with creative solutions, and finally a list of people that would also need to be included in the conversation. This continued until at the end of the meeting where everyone listed had already been met with.

The powerful part of this process was these influential people across campus became part of the solution. It included their ideas. They had been included in the design process. They had been consulted before funding was applied for. This was an indication of respect for them and their role in how the university ran. It showed interest in their ideas and experience. And it made them aware we understood it could not be done without their support. They were now part of the innovative new approach and buy in was high and widespread. This connection across campus continues to be valuable.

The other group that needed to be included before funding was who would be on the team. To design a modality and develop new curriculum we needed experienced faculty and an instructional designer. An email was sent to all faculty in the School of Computing. Four faculty responded. Along with the PI, we now had five faculty. The instructional designer for our college was contacted and accepted the invitation to join the project. This would form our core team.

And team-driven it was. All the decisions were made collaboratively. There were some high-level plans included in the funding application but everything was on the table. The name, the design, the details that would make it work, and the enthusiasm all came from the team. This was difficult for a PI that thought she had a great idea, something she had wanted to do for many years. To share those ideas and let the team mold, change, and create something new was both difficult and valuable. Collaboration produces significantly better outcomes and this was evident as the solution evolved.

Additional Reflections and Challenges of Collaborative Development

We have received significant feedback on several collaborative aspects of CS Flex. First, CS Flex faculty often report improved course quality due to the accountability of work in pairs for course development. They indicate they learn from one another during this process and incorporate new ideas both in their CS Flex courses and beyond.

In terms of student feedback, and specifically student feedback on collaborative group work, we have anecdotal evidence students are enjoying working on more highly structured and instructor-guided group work. This feedback has been in the form of student-faculty meetings that are integrated into the CS Flex courses, as well as on course reviews. Since students self-select into this modality, it is likely that the methods used and flexibility given are a match for some and that traditional online or face-to-face courses are a better match for others. CS Flex simply gives students another option and we allow students to decide which suits them best.

This doesn't mean that all feedback has been positive. Some faculty and staff express a reluctance to support this model. Like with the students, faculty are not required to participate in this modality. They self-select if this type of course development or instruction suits them. Some faculty elect to primarily contribute as course developers, others as course instructors, and many in both roles.

There is also a need for significant awareness of curriculum and technology changes happening across campus. CS Flex course updates take time and resources, and must be done quickly when changes occur. For example, when new requirements for adding Student Learning Objectives to our syllabi or using new exam management software for remote testing are required at the University, all courses must be updated quickly and in a common way agreed upon by the team.

Future Work

As we move forward with CS Flex, we don't currently anticipate major changes to most of the collaborative processes we have described here. As our design processes are iterative, we anticipate adjusting and improving them as we continue to learn more from our experiences and as we examine data we have been collecting about our offerings.

We have learned that growth can uncover the need for new approaches. We anticipate continued growth both within the department as the team grows and potentially across campus with other interested departments. As we have done in all of our development, we will be responsive to new needs and they will drive our future work as they arise.

Conclusion

In this paper, we have presented details on the process, successes, and challenges we have faced in this multifaceted collaborative approach to program design. Specifically, CS Flex has integrated collaboration in multiple ways to support flexible, mastery-based learning. This collaboration goes beyond traditional student group work. It integrates collaborative practices into course development and delivery, as well as program development and across campus integration.

While the CS Flex modality may not be itself reproducible or desirable in all contexts, the collaborative approaches described here may be integrated within many educational contexts. Practices including paired course development, shared templates and policies, clearly defined group work practices, and democratic decision making can be taken as they fit specific education contexts.

This collaboration has great potential and benefit, and comes with its unique set of challenges. For example, support for ongoing development takes buy-in and resources both from individuals and from the institution.

As we move forward with CS Flex, we will continue examining the impacts of the CS Flex model and these specific collaborative practices on student persistence and both technical and soft skill development. We will also continue to explore faculty satisfaction surrounding this model and its practices. While the CS Flex model is valuable in its own right, we also believe that these individual collaboration approaches may be valuable to consider for more general adoption. Regardless of the future of CS Flex, we believe that collaboration is a key skill of all technical practitioners and we will continue to explore practices to support mastery in this area.

References

[1] K. D. Fuez, L. DuHadway, N. Anderson, R. C. Fry and J. Christensen. “Lessons Learned Creating a Flexible Online Program” *2026 ASEE Annual Conference & Exposition*. 2026.

[2] S. Mason, “Collaborative learning in computing education: Faculty perspectives and practices,” in *Proc. Int. Comput. Educ. Res. Conf*. (ICER), Aug. 2020, pp. 136–146.

[3] S. Umapathy and A. D. Ritzhaupt, “A meta-analysis of pair programming in computer science education,” *ACM Trans. Comput. Educ*., vol. 17, no. 4, pp. 1–13, 2017.

[4] M. Veldman, J. Admiraal, and A. van Tartwijk, “Teacher collaboration for professional learning: A systematic review,” *Educ. Res. Rev*., vol. 29, 2020.

[5] K. D. Feuz, L. DuHadway, H. E. Valle, R. C. Fry, and K. M. Murphy, “Improving student learning through required exposure to other students’ code via discussion boards,” in *Proc 2020 ASEE Virtual Annu. Conf.*, 2020.

[5] N. Anderson and T. Gegg-Harrison, “Pair2 learning = pair programming × pair teaching,” in *Proc. 17th Western Canadian Conf. Computing Education*., 2012, pp. 2–6.