Lessons Learned Creating a Flexible Online Program

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. Linda Duhadway, Weber State University

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

Nicole Anderson, Weber State University

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

Julie Christensen, Weber 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

Lessons Learned Creating a Flexible Online Program

Abstract
Students enter college with diverse backgrounds and prior experiences. Some students require more time to understand and master the material, while other students can move through a course more quickly. We designed and developed courses that allow students to adjust due dates to accommodate these differences. When we first proposed an approach that supported individualized, flexible due dates, many agreed it was a good idea but thought it was impossible to implement. We have now been offering such courses for more than five years.

In this paper, we describe a new teaching modality (CS Flex) for higher education, designed to help students achieve learning outcomes while reducing barriers to student success. CS Flex allows students to leverage the expertise and skills they bring to the program and build on those abilities to complete it in a timeframe that fits their personal education and employment goals. This program has several interesting characteristics that, when combined, form a truly unique program.

CS Flex blurs the boundaries of the traditional semester, offering open entry and early exit opportunities. This enables continuous enrollment and progression in the program as soon as they are ready, creating a schedule that works around the individual student rather than around a course or a semester. This provides students with the opportunity to start a CS course when they are ready, without waiting for the traditional semester. The flexibility in pacing and start dates allows students to complete a course and continue in the program at an accelerated pace.

The CS Flex program includes additional elements based on pedagogical best practices, which further strengthen the program. Each course is divided into modular learning units and mapped to explicit learning outcomes. Modules include consistent elements such as readings, interactive materials, formative assessments, and summative mastery checks that ensure transparency and predictability across courses. Progression depends on demonstrated mastery, reinforcing depth of understanding rather than seat time.

The CS Flex program is designed to achieve the same learning objectives as standard CS courses while offering a distinct learning experience. Course topics and materials align fully with the core CS curriculum, allowing students to combine CS Flex and traditional CS courses to meet degree requirements. Each module in a CS Flex course is structured around clearly defined learning outcomes. This reinforces a consistent and outcome-driven learning experience.

Experienced CS faculty develop CS Flex courses to provide a carefully structured online learning experience that maintains academic rigor while supporting flexibility in scheduling. Instructional materials are designed and developed in advance so instructors can focus on meaningful student interactions. Consistent student-teacher interaction and rapid feedback are integral parts of the curriculum design.

This paper examines the design, implementation, and evolution of the CS Flex program, presenting lessons learned from five years of practice. We discuss how design choices, such as modular structure, mastery progression, and rapid-feedback pedagogy interact to support student success. Significant lessons were learned and we found that implementing structured limits and track shift policies improved completion rates and decreased the average time to completion.

Introduction

CS Flex is an online course modality built on five core principles: Individualized flexible due dates, Mastery-based progression, Collaborative design, Student success, and Facilitating group work. CS Flex was created to better support a diversity of student backgrounds and needs within the constraints of a university setting.

University computer science courses serve a wide and diverse student population with varying degrees of prior knowledge [1]. Students range from those with no prior programming experience to those who have already worked in the field. Some students demonstrate strong aptitude and progress quickly, while others require additional time and practice to develop understanding. Students' motivations for successful learning also vary: some are driven by interest in the discipline, others by career prospects, and still others by external personal or professional requirements [2, 3]. In addition, some students focus primarily on their coursework, while others balance significant family and work obligations [4]. In a traditional classroom setting, it is difficult to effectively meet the needs of such a diverse population.

At the same time, universities operate within fixed structural constraints. Semesters begin on specific dates and last a predetermined number of weeks. Courses are defined by required topics, assignments, and assessments that students must complete to earn credit. In traditional course formats, instructional material is presented to all students at the same pace, and students are generally expected to progress through the material simultaneously, regardless of individual preparation, learning speed, or external circumstances.

The CS Flex program has gone through multiple iterations of improvement throughout its existence. In the initial planning phases we had some ideas about the goals we wanted to achieve and ways the program could support those goals. We established five CS Flex design principles. We spent the first year piloting those ideas and continuing to improve and refine the details. Since then, we have continued to make changes to the program based on input from faculty and students involved in the program. The remainder of this paper will discuss each design principle in more detail including how we have continued to refine the policies and processes supporting those principles. We present data demonstrating the effectiveness of the program and the impact policy changes have had on student outcomes.

CS Flex program details

A. Individualized flexible due dates

CS Flex courses are designed to allow students to move through course material at an individualized pace, but within intentionally defined limits. While some flexible programs eliminate due dates entirely [5], such approaches can lead to unintended consequences, including procrastination [6], last-minute cramming, and lower-quality learning outcomes. In some cases, students delay work until completion becomes infeasible, resulting in high non-completion rates. CS Flex seeks to avoid these outcomes by balancing flexibility with structure.To achieve a balance between flexibility and structure, CS Flex courses retain structured due dates while providing controlled mechanisms for adjusting those deadlines. These limits are deliberately designed to support sustained engagement [7], maintain academic rigor, and ensure appropriate

progression through prerequisite material. When due dates are shifted, they are shifted by one week. Each shift is called a track shift.

To achieve flexibility while still enforcing due dates we created some tools to interact with the learning management system (LMS) through its api. These tools allowed us to create a standard set of due dates for all the assignments in the course and then apply the due dates individually to each student. When a student needs to accelerate or decelerate in the course their due dates can be adjusted accordingly.

Across several iterations of the track shift policy we have settled on the current policy. The standard track due dates are set to a slightly accelerated pace and have students completing the fifteen week course in approximately twelve weeks. A student can request an unlimited number of acceleration track shifts provided their next due date is at least one week out. This ensures that they do not accelerate past any unsubmitted due dates. For decelerating, students start the course with a total of eight deceleration track shifts that can be used throughout the course. Three of these can be used at the student's discretion, as long as the track shift request is made 24 hours in advance of their next due date. The remaining track shifts are applied by the instructor when a student fails to demonstrate mastery on a challenge activity. If a student has used all eight track shifts they are no longer able to decelerate in the course.

We also have a policy that allows students to pause a course. The idea of a course pause is that sometimes things come up, both planned and unplanned. A student may have a medical emergency, a family vacation, or a busy couple of weeks at work. A course pause allows them to address the situation without having to complete the coursework during that time. We allow a single course pause per course per semester. Course pauses may not be taken during module 0 or module 1 of the course. If a student needs to pause the course at that point, they should instead start it at a later date. A course pause may last for one to three weeks as requested by the student. Each week counts toward the total of eight deceleration track shifts.

With these policies in place, a student can complete a course very quickly and is guaranteed to be done with the course (successfully or not) within twenty weeks of starting the course: twelve weeks plus eight weeks of deceleration. Students who complete a class before the semester ends may move on to the next class. Those who are still working in the class at the end of the semester are given a D-. Then, once they complete the coursework, the grade is changed to the grade the student has earned. This approach was decided in conjunction with experts from the registrar's office, financial aid, and advising.

Students use this flexibility in different ways. Some take advantage of CS Flex to get a strong start at the beginning of the semester, while others aim to complete a course early in order to focus on additional coursework later in the term. Other students intentionally progress more slowly, using the flexibility to spend additional time on challenging topics in order to achieve mastery of the material. For students seeking admission into our master’s degree program, CS Flex can be particularly beneficial for completing required leveling courses. These courses are typically sequential, with each course serving as a prerequisite for the next. In a traditional format, completing the sequence can require multiple semesters; with CS Flex, motivated students may complete the sequence within a single semester, allowing earlier entry into graduate-level coursework.

One primary purpose of CS Flex is to provide opportunities for acceleration. Some students are able to progress more quickly through learning materials and benefit from the option to complete a course in less time. While CS Flex and some online courses allow early completion, finishing a course ahead of schedule has limited value if students must wait until the next semester to enroll in the subsequent course. True acceleration occurs only when students can begin the next course immediately after completing the previous one. Implementing this form of acceleration required structural changes that introduced new administrative and instructional challenges.

To address this need, CS Flex allows students to start courses during the semester rather than only at the traditional start date. This is implemented through second-block course sections. A student who completes one course early may register for the second-block section of the subsequent course and begin without delay. However, these rolling start points introduce operational complexity, including the need for close coordination among advising, registration, and faculty scheduling. Enrollment in second-block sections is also inherently uncertain, as demand depends on when students complete prerequisite courses.

These second-block sections are intentionally small and are administratively paired with a full-block section for faculty workload and compensation purposes. While this pairing enables the program to offer flexible start times without incurring additional instructional cost, it requires careful planning and monitoring to ensure that faculty workload remains balanced and that students receive timely guidance as they transition between courses.

B. Mastery-based progression

In CS Flex, a student's progression through the modules and thus through the course is dependent on their ability to complete the summative mastery check at the end of each module. The form of these summative assessments is particular to each course and can be adjusted to fit the learning objectives and requirements of the course. In some modules it may take the form of a programming assignment or project. In other courses it may take the form of a quiz, project deliverable or other artifact. The core concept of these mastery checks is that a student should not move on to the next module before they have demonstrated mastery of the current module. At the same time a student's overall progress in the course and in the program should not be unduly hampered if they need to spend an extra week or two to fully understand a complex topic. The CS Flex program achieves both of these goals by enforcing a summative mastery check before moving on to the next module and providing the ability to shift due dates if the mastery check is not successful on the current attempt.

When faculty know in advance that students will have demonstrated mastery on all the prior modules it provides additional opportunities when designing the course. Content in the modules can be built to depend upon the content from previous modules and faculty have confidence that students successfully completed those modules if they are at this point in the course.

Each module is designed around clearly defined learning outcomes. Activities, assignments and quizzes in the module are directly tied to these learning outcomes. Not all learning outcomes for a module are required to be addressed by the summative assessment but all learning outcomes for the module are required to be reflected in one or more activities of the module.

Requiring students to demonstrate mastery of the topic before moving on to the next module has also created its own set of challenges. For example, modules must be designed in such a way that we can measure whether a student has demonstrated mastery of the topic. This is not a novel requirement in higher education and designing content with measurable student outcomes has long been considered best practice. The difference here is that it is enforced by the structure of the program. You cannot have a mastery component to every module without having one or more measurable outcomes.

Recently, we have updated the threshold for mastery on exams. On challenge activities the threshold is set to an 80% score on the activity. If a student scores less than an 80% on the activity they will be given feedback and a track shift and will have the opportunity to resubmit and demonstrate mastery. On exams the threshold has been set to 70% but students only get a single retake to demonstrate mastery. The retake exam is a new exam with new questions so students cannot simply memorize the answers and retake the exam. This 70% threshold better aligns with our traditional courses and allows students to continue to progress in the course when they have demonstrated sufficient understanding of the material up to that point.

C. Collaborative design

Each CS Flex course is co-developed by two experienced faculty members. These faculty developers meet regularly to collaboratively design the course, divide development tasks, and review one another's work. Unlike the traditional model in which a single faculty member independently owns and develops a course, this approach emphasizes shared ownership, peer review, and collective responsibility for course quality. For the faculty involved, this represented a departure from common practice. While collaborative development introduced new challenges, it also created opportunities for shared reflection, increased consistency, and improved course quality. Additional discussion of this collaborative approach is provided in [8].

During development, course developers also meet regularly with the CS Flex Coordinator. The coordinator serves as a sounding board for instructional and logistical decisions, drawing on experiences from other development teams and ensuring alignment with CS Flex expectations. The coordinator also reviews completed courses to verify consistency with program goals and standards.

To keep course materials current and relevant, each CS Flex course is scheduled for a major update every three years. For each update, a development team is formed that may include previous course developers, faculty who have taught the course, or additional faculty with topic expertise. The same collaborative development process used for initial course creation is applied to these updates, reinforcing consistency and enabling systematic improvement over time. This scheduled revision cycle mirrors maintenance and refactoring practices in software development, ensuring that course content evolves in response to changes in the discipline and instructional context.

As CS Flex courses are developed, the team emphasizes pedagogical best practices for effective online instruction. Given the scale, flexibility, and consistency required by the CS Flex modality, the program adopted a structured and collaborative course development model. The Instructional

Design Team at Weber State University provides faculty with consultation services and maintains a current checklist of online teaching best practices. This checklist is referenced throughout CS Flex course development. In addition, an instructional designer is included as a member of the CS Flex team and provides guidance on implementing effective design strategies during course creation.

The instructional designer contributes not only during course development, but also to the formulation of program-level policies. This role supports consistency across courses and provides a design-informed perspective when evaluating proposed changes and recommending improvements.

One of the early implementation decisions in CS Flex was to adopt common design elements across courses. Because multiple courses were being developed, the team anticipated benefits from consistent organization and delivery. A standardized course structure allows students to become familiar with navigation patterns and instructional expectations, reducing cognitive overhead and helping students feel confident moving through the course. This consistency provides a framework that supports student success as they progress through multiple courses within the modality.

CS Flex courses are organized into modules, each of which begins with an overview that introduces the module and identifies the learning objectives. Modules contain a series of structured activities designed to support learning and mastery:

- **Learning Activities:** These activities introduce topics and provide opportunities to practice skills. They are lightly graded and emphasize learning through doing, rather than simply observing. Completion is highly recommended to support effective skill acquisition.

- **Try-It-Out Activities:** These activities offer guided practice opportunities to reinforce the skills being learned. The grading is focused on completion, allowing students to improve their understanding while boosting their score.

- **Challenge Activities:** These summative activities are designed to demonstrate mastery of the module's learning objectives. The majority of module points come from challenge activities, which require a score of 80% or higher for mastery. Successfully completing challenge activities unlocks the next module.

This modular structure, with clearly defined requirements for completion and a predictable pattern across courses, promotes transparency, consistency, and student autonomy. Students know what to expect in each module and can focus on mastering the content at their own pace within the boundaries of course deadlines. By integrating structured learning, practice, and mastery opportunities in every module, CS Flex ensures that students engage with material iteratively and systematically, reinforcing both skill development and progression toward course objectives.

Overall, incorporating practices drawn from software engineering—such as collaborative development, scheduled maintenance, peer review, and the inclusion of specialized

expertise—has improved the quality, consistency, and sustainability of CS Flex courses. Although there was initial hesitation in deviating from traditional individual course development, faculty reported improvements in course design and significant value in the sharing of educational and instructional experiences.

D. Supporting students

A core principle of CS Flex is supporting students through intentional instructional design and timely interaction. To implement this principle consistently across courses, CS Flex establishes explicit expectations for faculty regarding availability, responsiveness, and feedback. Course materials are fully developed before the class begins. This allows faculty to focus on student support and interaction rather than creating instructional content during course delivery.

Each course includes an early-semester activity that requires direct interaction between students and the instructor. While instructors are given flexibility in how this interaction is implemented, the expectation that early contact occurs is consistent across all CS Flex courses. Activities may include individual meetings, asynchronous video interactions, or personalized written communication. This early engagement helps establish expectations, build rapport, and identify potential student needs.

Student success is further supported through an emphasis on rapid feedback. Instructors are expected to respond promptly to student correspondence and to provide grading and feedback on submitted assignments within one to two business days. This quick feedback enables students who demonstrate mastery to progress without unnecessary delay and encourages sustained engagement with the course material. When submitted work does not meet mastery expectations, two things happen. The instructor provides detailed feedback so they know what worked and what did not. This gives students direction in how to achieve mastery. A track shift is applied, which extends the due dates by one week. This provides time for students to incorporate feedback and achieve mastery while maintaining forward momentum in the course.

An academic coach helps students throughout their entire program. Initially, the CS Flex coach assisted the faculty in setting up their courses by manually configuring the tracks. This was time-consuming, and after a short period, it was added to the Flex tools LMS integration developed for supporting CS Flex. The academic coach was present in every CS Flex course to assist with track shifts during the first several years, supporting the faculty. However, this task was eventually dissolved to allow the coach to focus more on the students than on the course logistics and the Flex tools LMS integration was further extended to support enforcing track shift policies. The coach works closely with the CS Flex Coordinator and faculty, discussing when classes are full, marketing needs, or student admissions and registration. Working closely with the admissions, financial aid, and registrar’s offices is also vital to students' success.

The academic coach meets with students individually to introduce them to CS Flex. Most students have limited knowledge of the program and are eager to learn about its benefits. They are excited about the program and how it will help them in their educational goals. Meeting with every student before admitting them to CS Flex helps retain students by answering their questions one-on-one.

Initially, admissions to CS Flex consisted of four steps:

1. Be admitted to Weber State University
2. Complete the CS Flex Application
3. Attend the CS Flex Information Session
4. Complete the Interest Form

In Step 3, students would attend the CS Flex Information Session by having a 30-minute one-on-one meeting with the academic coach. This is where the students learned how CS Flex was structured and how it differed from other online classes. The most common student questions were about acceleration and deceleration. The information session was positive, and many students asked why we didn't teach more classes outside of computer science this way. After the four steps are completed, the academic coach will add the students to the CS Flex cohort and send each student an email welcoming them to CS Flex, letting them know they can now register.

Recently, the admissions steps to CS Flex changed slightly. This is the new format:

1. Be Admitted to Weber State University
2. Learn CS Flex Policies (Watch an 8-minute video)
3. Pass the Quiz with 80% on the video training
4. Complete the CS Flex Application
5. Meet with the Academic Coach to answer questions and complete the Interest Form

Supporting students throughout the program includes helping them register for classes, answering questions about tracks and start dates, and providing guidance on course selection. The coach is also an academic advisor and will advise students on their class schedule, discuss their major, and support their questions as needed. After about two years, if students are no longer taking CS Flex courses, the student is removed from the CS Flex cohort. Scholarships are given to K-12 teachers who want to take CS Flex courses. The teachers apply the same way as regular students and meet with the academic coach to complete the scholarship application.

E. Facilitating group work

One of the key challenges that arose during the development of our sophomore software engineering course, the first CS Flex course with a significant groupwork component. In this course, students not only learn individual software engineering skills but are also required to apply their knowledge collaboratively on a substantial team project. Early on, there was concern about whether a course could effectively integrate teamwork while maintaining individualized, flexible due dates. Addressing this issue became another implementation hurdle for the CS Flex team, which we approached using our established process of team brainstorming, discussion, and collaborative decision-making.

The course structure was designed with the first five modules focusing on foundational software engineering concepts, followed by an exam. These modules were completely individually. Module 6 introduced agile development principles and strategies for effective team collaboration. The group project was divided into three modules, modeled after industry-style sprints, with tasks distributed among three to four team members. This approach not only simulated real-world software engineering workflows but also reinforced iterative skill development and peer accountability.

Coordinating individualized schedules with team deadlines required a structured solution. Module 6 included a “ready-for-team” assignment; once students completed it, the instructor was notified, and the student was placed on hold—pausing all future due dates—until enough students reached the milestone to form a team. Once the team was assembled, unified due dates were established for all members, with the option for teams to request future adjustments. This process ensured that students could experience authentic agile teamwork while still progressing at their own pace, fostering both collaborative skills and mastery of individual software development competencies.

By carefully designing modular sprints and synchronizing team milestones, CS Flex successfully integrated teamwork into a flexible, individualized course model. Students benefited from a combination of personalized pacing, structured collaboration, and exposure to industry-relevant agile practices, demonstrating that flexibility and team-based learning can coexist effectively.

Since then, we have added the advanced software engineering course, which follows a similar structure. Through experience, we have identified approaches to navigating exceptions arising from the combination of individual pacing and groupwork. For example, a student who progresses slowly through the individual modules may reach the “ready-for-team” assignment after all other students have been assigned to teams. In this case, the student continues in the course past the end of the semester and is assigned to the first team formed in the following semester. Other exceptions include allowing a student to shift teams when an individual pause in progress is required. We anticipate that unique situations will continue to arise, and individualized solutions are developed collaboratively among the student, the instructor, and the CS Flex Coordinator to maintain both flexibility and alignment with course objectives.

Results

The CS Flex program, with the innovations described above, has been successful across several metrics. To quantify the program's success, we have gathered course data for the entire CS Flex program from Fall 2020 to Fall 2024. Summary statistics are shown in Table 1. We worked with our institutional data representative to collect students' course enrollments, final grades, and the dates of their first and last submissions per course. All data collected was anonymized by the institutional data representatives prior to being provided. Additionally, we received Institutional Review Board (IRB) approval to conduct this study.

Total Enrollments (all modalities)	13639
Total Enrollments (CS Flex only)	4139
Total Unique Students (all modalities)	3680
Total Unique Students (CS Flex only)	1292

Table 1. Summary of Enrollment Data from Fall 2020 to Summer 2024

First, we consider the successful course completion rates. A student is considered to have successfully completed a course if they earned a final grade of C or higher. We use this threshold because the CS program requires a C or higher in all CS courses. Figure 1 shows the successful course completion rates by semester. Using an unpaired t-test, there was no statistically

significant difference in course completion rates between the CS Flex modality and all other modalities the CS program uses (face-to-face, hybrid, and online).

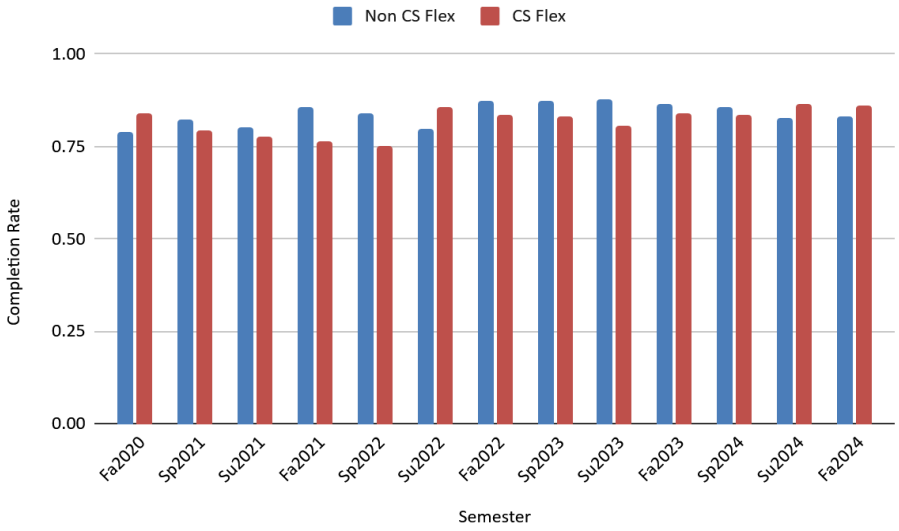

Figure 1: Course Completion Rate by Semester. Differences in course completion rates between CS Flex classes and Non CS Flex classes are not statistically significant.

We are also interested in how quickly a student completes the course. We calculate the time spent on the course by counting the number of days between a student's first submission and last submission to the LMS. For CS Flex courses, this is an accurate gauge of the number of days a student spends in the course, since the course starts with a syllabus quiz due within the first few days and ends with a final summative assessment submission. Unfortunately, the same is not true of other CS courses taught in different modalities, as not all of them have a consistent required submission at the start of the course. For this reason, we only consider CS Flex classes when calculating time spent in the course.

Figure 2 shows the average time spent on a CS Flex course by semester. As can be seen, students on average are spending approximately fifteen weeks per course, which is in line with a traditional fifteen-week semester course.

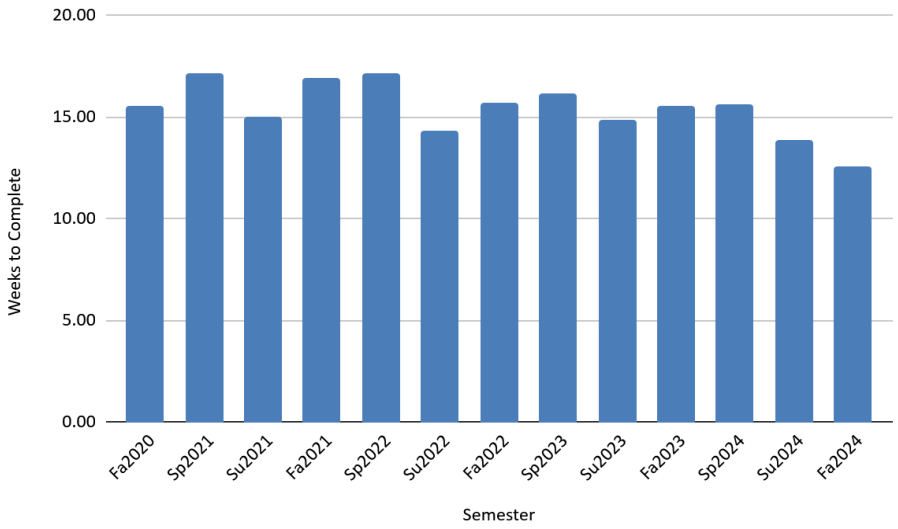

Figure 2: Average time to successfully complete a CS Flex course by semester. There is a downward trend in completion time as we continue to refine and improve the CS Flex program and its associated policies.

Looking only at the average time to completion hides the benefit some students see by choosing to accelerate through the CS Flex program. Figure 3 shows the minimum time spent on a CS Flex course by semester. Clearly, each semester, some students are making use of the flexible pacing of the CS Flex program to complete their courses at a much faster pace than the traditional semester schedule would allow.

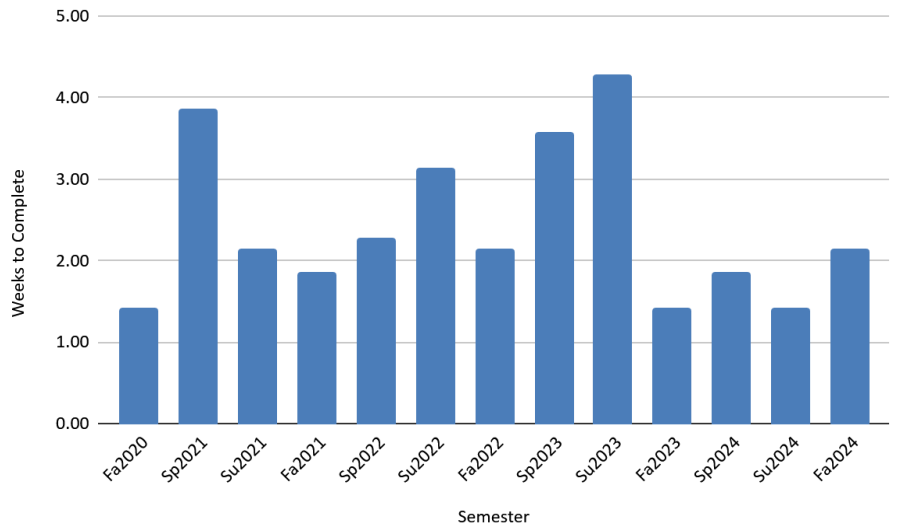

Figure 3: Minimum time to successfully complete a CS Flex course by semester. Each semester there are at least some students who use CS Flex courses to significantly accelerate their pace through the program.

Finally, it is interesting to note that during the timeframe of this study (Fall 2021 to Summer 2025), we have had 501 students successfully complete a CS Flex course in less than twelve weeks, which is 15.5% of students successfully completing a CS Flex course.

Discussion

During the initial design period, the CS Flex Team consisted of five faculty members, an academic advisor, and an instructional designer. Over time, the team grew to its current size of nineteen members. The CS Flex Team meets monthly during the academic year. These meetings are to keep an open discussion about CS Flex. Ideas are presented, concerns are shared, and changes are discussed. Once a consensus is reached, changes are made. This ensures that we have collective buy-in on the outcome based on the discussion process and a majority vote. When a proposal has some support but we cannot reach a consensus on the outcome, it is researched and discussed in future meetings. Design, development, and improvement of CS Flex are ongoing and remain team-driven.

In the pilot program, we gave students a large amount of flexibility, allowing a student to request a track shift once per module, with a total of ten track shifts available to the student. The pilot program showed that with good course design, proper tooling, and strong partnerships across the university, students could succeed in this flexible learning environment.

The pilot program also highlighted some weaknesses of the program structure. Although acceleration was the goal, many students were instead using the flexible pacing of the course to slow down. Deceleration does not have to be problematic and in many cases has a positive benefit as students slow down and take the time to fully master one topic before moving on to the next. However, some students were taking advantage of the deceleration policies to unnecessarily impede their own ability to complete the course in a timely manner and move on to the next course in the degree program. Our experience showed there was a need to place additional restrictions on the flexibility provided to students. We implemented the change in track shift policy beginning in the Summer 2022 semester.

This change has impacted student course completion rates and the time it takes students to complete a course. When comparing the average course completion rate from Fall 2020 until Spring 2022 vs the average course completion rate from Summer 2022 until Fall 2024 there is a clear increase in course completion rates. The difference in means between these two groups is statistically significant using an unpaired t-test with a p-value of 0.003. Similarly, when comparing the average time to complete a course during these two time periods the difference in means between groups is statistically significant with a p-value 0.039. This demonstrates the importance of maintaining structured due dates in a course and providing bounds on due date flexibility.

Over time we have also made changes to better align course content with course learning objectives. It is now a requirement that each question on every exam and quiz be tied to a specific learning outcome for that course. Going through that process has improved the quality and consistency of our exams and quizzes across all the courses. We have also updated the mastery threshold for exams. This change was made in Summer 2024. With these changes and other improvements to CS Flex policies and courses we see an upward trend in course completion rates over the last several semesters and a downward trend in time spent to complete a course. This means that on average more students are successfully completing courses and completing courses at a faster pace than they were when we first started the program.

Related Work

Many other software engineering and computer science programs have explored ways to enable additional flexibility for students attempting to complete a course (e.g. [9,10]). [11] considered the effect of deadlines in an online self-paced course. Over several semesters, they altered the deadline policy in a single course from no deadlines, to suggested deadlines, to deadlines with minor penalties, and finally deadlines with minor penalties and a hard mid-term deadline to complete a certain number of assignments. They found that deadlines (even just suggested deadlines) help students manage their time better and better assess their likelihood of passing the course.

Offut el. al. [5] created a CS1 programming course without fixed deadlines and also had a process in place for students to finish the course up to 10 weeks after the end of the semester. They found that 10-25% of students use the additional time beyond the end of the semester to successfully complete the course.

Feuz et. al. [12] demonstrated novel ways for student-student interaction and code reviews when students are moving through a course at different paces. This was an early phase of the pilot program for CS Flex and helped demonstrate that student-student interaction is still possible and effective even when students are working asynchronously and at different paces as they move through the course.

Using a different approach, [13] looked at the introductory programming course sequence and designed some additional flexibility in how students progress through the sequence of courses. Each student takes the same CS1 programming course. Based on their performance in that course they are then recommended to either move on to the standard CS2 course or to take a slower-paced two semester course that reinforces the concepts of CS1 in addition to also covering the CS2 material. They found that the slower-paced course was beneficial for students who opted into that course.

Future Work

The evolution of the CS Flex program is an ongoing process driven by data and faculty collaboration. As we move forward, several key areas for research and programmatic refinement remain. In Summer 2024, the mastery threshold for exams was adjusted to 70% to better align with traditional course standards. Future research should specifically isolate this variable to determine if the lower threshold impacts student performance in subsequent, more advanced modules or courses.

A critical metric for any new modality is how well it prepares students for the next level of study. We plan to conduct a longitudinal study comparing the performance of CS Flex students in traditional upper-division courses against their peers who followed face-to-face or standard online tracks. Additionally, while 15.5% of students complete courses in under twelve weeks, others utilize the full twenty-week window allowed by deceleration and the end-of-semester "D-" policy. We intend to analyze the correlation between completion timing—specifically comparing

those who finish early, on time, or late—and their performance metrics in the course and subsequent courses.

Finally, the implications of AI on mastery-based assessments must be addressed. We plan to explore how generative AI tools might be integrated into the "Try-It-Out" activities while ensuring the summative "Mastery Checks" remain rigorous and authentic.

Conclusion

The CS Flex program was developed to bridge the gap between the rigid structure of traditional higher education and the diverse needs of modern computer science students. By centering the modality on five core principles—individualized flexible due dates, mastery-based progression, collaborative design, student success, and facilitating group work—we have created a system that prioritizes depth of understanding over simple seat time.

Our data from Fall 2020 to Fall 2024 demonstrates that CS Flex is a viable and effective alternative to traditional modalities. While completion rates are statistically comparable to face-to-face and standard online courses, the program offers unique benefits in terms of acceleration and individualized pacing. Significant lessons were learned during the pilot phases, most notably that "boundless" flexibility can lead to counterproductive deceleration; implementing structured limits and track shift policies actually improved completion rates and decreased the average time to completion.

Ultimately, CS Flex demonstrates that with intentional design, robust instructional support, and specialized administrative tools, universities can offer a learning experience that is both flexible and academically rigorous. As we continue to refine these policies, the program stands as a model for creating student-centered pathways in technical education

References

[1]. R Ball., L. Duhadway, K. Feuz, J. Jensen, B. Rague, and D. Weidman. “Applying machine learning to improve curriculum design,” In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education,* 2019 pp. 787-793.

[2]. J. Pirker, M. Riffnaller-Schiefer, and C. Gütl. “Motivational active learning: engaging university students in computer science education”. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* 2014. pp. 297-302.

[3]. D. López-Fernández, E. Tovar, L. Raya, F. Marzal, and J. García. “Motivation of computer science students at universities organized around small groups,” *IEEE Global Engineering Education Conference (EDUCON)*, Dubai, United Arab Emirates, 2019 pp. 1120-1127

[4]. F. Rahman, “Understanding Barriers and Motivations of Non-Traditional Students Learning Programming in an Online CS1 Course,” In *Proceedings of the 21st Annual Conference on Information Technology Education* 2020. pp. 38-41

[5]. J. Offutt, P. Ammann, K. Dobolyi, C. Kauffmann, J. Lester, U. Praphamontripong, H. Rangwala, S. Setia, P. Wang, and L. White. "A novel self-paced model for teaching programming," *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*. 2017.

[6]. N. Kooren, C. Van Nooijen, and F. Paas. "The influence of active and passive procrastination on academic performance: A meta-analysis," *Education Sciences* 14.3 2024..

[7]. J. Lim, "Predicting successful completion using student delay indicators in undergraduate self-paced online courses." *Distance Education* 37.3 2016 pp. 317-332.

[8]. N. Anderson, L. DuHadway, K. Feuz, R. Fry, and J. Christensen. “Developing a Responsive Computer Science Program with a Collaborative Design Model” 2026 ASEE Annual Conference & Exposition. 2026 [submitted]

[9]. F. Castro, J. Leinonen, and A. Hellas. "Experiences with and lessons learned on deadlines and submission behavior." *Proceedings of the 22nd Koli Calling International Conference on Computing Education Research*. 2022.

[10]. G. Gill and C. Holton. "A self-paced introductory programming course." *Journal of Information Technology Education: Research* 5.1 2006 pp. 95-105.

[11]. M. Chiu, E. Moss, and T. Richards. "Effect of deadlines on student submission timelines and success in a fully-online self-paced course." *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*. 2024.

[12]. K. Feuz, L. DuHadway, H. Valle, R. Fry, and K. Murphy "Improving Student Learning Through Required Exposure to Other Students' Code via Discussion Boards." *2020 ASEE Virtual Annual Conference Content Access*. 2020.

[13]. K. Whittington, D. Bills, and L. Hill. "Implementation of alternative pacing in an introductory programming sequence." *Proceedings of the 4th conference on information technology curriculum*. 2003.