Accommodating Shortened Term Lengths in a Capstone Course using Minimally Viable Prototypes

Nathan Eloe
School of Computer Science and Information Systems
Northwest Missouri State University
Maryville, MO, USA
nathane@nwmissouri.edu

Charles Hoot
School of Computer Science and Information Systems
Northwest Missouri State University
Maryville, MO, USA
hoot@nwmissouri.edu

***Abstract*—This Full Paper presents a novel way of implementing the first half of a two-semester software design and implementation capstone course. The paper compares prior work from traditional (15-week) terms to a compressed (7-week) format and a significantly compressed (4-week) format employing this approach. To maximize competitiveness, many programs are under pressure to do more with less: competing to provide more outcomes and more practical real-world experiences in less time and with fewer courses. These challenges become especially significant in capstone courses, where students, still learning, must design and implement solutions consolidating a wide variety of skills and knowledge covered in the curriculum.**

The capstone course spans two semesters; the first semester focuses on design and the second semester focuses on implementation. In the standard approach, teams meet with the client and develop design documents including use cases, user interface sketches, and data models. In this approach, rather than starting from scratch, students received artifacts from a previous capstone course employing a consultancy model. Artifacts included a request for proposal (RFP), a minimally viable prototype (MVP) partial implementation, and corresponding design documents. The work reviews students' ability to effectively incorporate the provided artifacts to produce a high quality design for complete implementation of the project.

The work includes an evaluation of first-semester team deliverables generated under both the compressed and significantly compressed schedules and relates the expectations and outcomes to those in a traditional-length term. This approach offers a common means to assess schedule impacts and tailor the course in accordance with the relative concentration of contact hours associated with the schedule. This paper describes variations in expectations and options for MVP, client interaction schedules (including frequency and duration), and discusses the associated impacts as perceived by clients and mentors on engagement and student learning outcomes. Finally, the paper offers specific recommendations and lessons learned for those educators that are – or plan to – offer capstone courses under a highly compressed schedule.

***Index Terms*—computer science education, software engineering, design methodology, minimally viable prototypes, capstone, pedagogy**

I. Introduction

A. Related Work

Project-oriented capstone courses have been a critical part of education since before the 1990s [1], [2]. Capstone courses are motivated by many factors. For example, courses that involve students in real-world open source projects are found to increase their interest, improve their skills, and improve self-perceptions for a variety of capabilities [3]. Capstone projects may involve mentors, communities, businesses, or non-profits acting as clients and request student assistance with various initiatives [4]–[7].

One challenge of such courses is the relatively short calendar time and contact hours available for the project. Compared to a 40-hour workweek, any one course is likely to meet for about three contact hours, typically 50 minutes each for a total weekly in class time of 150 minutes. Typical terms span approximately 15 calendar weeks, a time frame considered short for teaching, interacting with the client, proposing, designing, implementing, and testing a viable application [8].

Capstone courses are also notable for bringing additional special challenges [9]. Factors that especially affect success include having a well-defined scope,the ability of students to work together, and how effectively the students engage with their clients [10]. To address some of the known challenges, work has been done to improve outcomes by incorporating professional development practices including test-driven development (TDD) [11], the use of tools such as Git for distributed version control [12]–[14], the application of consultancy models that mimic the request-for-proposal project process frequently encountered in industry [15], and incorporating formal contest models such as that used by Microsoft Imagine Cup Software Design Invitational (SDI) [16].

B. Motivation

As course offerings have evolved, the availability of shorter summer terms offers benefits for students. Students meet more frequently and for longer periods, and complete a course within approximately 7 weeks. Students meet five days each week, more closely approximating the typical 5-day week common in industry, with contact hours totalling six hours for each of the 7 weeks.

Recently, the first semester of this graduate level capstone course was scheduled to run over 4 weeks during the summer. Students met for 4 days each week and contact time was increased. This approach offered less calendar time to both the students and the instructors.

Issues identified in previous research include having a clear project scope, facilitating well-functioning teams, and fostering client engagement. To address these issues students were provided with two minimally-viable prototypes (MVP) as a starting point. These MVPs were developed by previous groups of students using an agile, consultancy-based model during their first semester capstone course. MVPs are commonly used in industry to assist with client communication, agile development practices, and can assist with avoiding larger design errors when following more formal design approaches [17], [18].

The goal was to jump start the design process and allow the students to accomplish a comparable set of learning outcomes and be just as prepared for the second semester of the capstone course as those students in a longer design course. Ideally the students would be able to get a clearer picture of their design when provided with a not fully realized prototype. The results from semesters of varying lengths give valuable insight into how to deal with shifting curriculum needs and educational time frames.

II. Methodology

A. Course Structure

1) 7- and 4- Week Sections: In a full length semester the expectation is that students in the first semester capstone will go through multiple iterations of their design and start the implementation of their solutions.

When one author was scheduled to teach two sections of the first semester capstone course over a seven week summer session, the expectations were adjusted. Fewer iterations were performed on each design artifact (at most three, with quick turn-around on feedback) and a lot of emphasis was placed on conceptualizing the problem before attempting a solution design. Graded project outcomes included:

- Problem Statement
- Functional and Non-functional Requirements
- Use Cases/User Stories
- Data Model (ER Diagram or Schema)
- UI Sketches and/or Mockups
- Proof of Concept Implementations
- First Iteration Plan

Each of these outcomes went through multiple iterations and drafts, with feedback being incorporated into the final product. The Proof of Concept implementations were designed to have students attempt to identify an appropriate set of tools/libraries to use for their project and show that they could implement a small non-trivial part of the system.

When the summer course needed to be shortened to four weeks the graded project the outcomes were simplified further:

- Functional Requirements
- Use Cases/User Stories
- Data Model (ER Diagram or Schema)
- UI Sketches and/or Mockups
- First Iteration Plan

Fewer iterations were done on each project artifact, and feedback was more direct. In addition, the students were provided with the design documentation and MVPs from a previous semester's class. The provided design was neither complete or perfect as it was focused on a critical vertical slice of the project. The hope was that with this base, a high quality design could be achieved with the limited additional iterations. These shortened courses' structures were similar to the Consultancy Model [15], but could not realize a full MVP in the shortened time frame without sacrificing most or all of the opportunity for design.

2) 15 Week Section: Recently another instructor was given the task of teaching a full length GDP and was asking for guidance for assignments and outcomes. The course structure for the 7 week version was provided as a starting point. While neither author was the mentor for this section of the capstone course, the final graded artifacts were roughly the same as those in the shortened summer sessions. Differences in instructional and mentoring styles resulted in differing formats and emphases on the artifacts, but the same set of deliverables allowed a relatively close comparison between the shortened and regular length semesters.

B. Artifact Rating System

Assessing team-based projects and the individuals within each team is challenging [10]. To simplify the comparison of the outcomes of all versions of the course, only the design submission of the groups as a whole were considered. The resulting design for each project was evaluated for the following artifacts:

- Data Model (ER Diagram or Schema)
- UI Sketches and/or Mockups
- Functional Requirements
- Use Cases/User Stories
- First Iteration Plan

In each area, two integer scores from 0 to 5 were awarded by a panel of the authors and indicate how well the design compared with a model design for that particular project (mentor expectations). The first score measures how complete the resulting design documentation is relative to a model design. For example, a set of Functional Requirements that encompassed all or most all aspects of the project as specified by the client would be awarded a score of 5, while an artifact missing a few major portions of the project would receive a score of 3.

The second score measures the quality of the design documentation. This score encompasses the student groups' ability to comply with the expectations set out in the assignment, the consistency in appearance of the documentation, and the readability and organization of the artifacts. This score, while more subjective than the first score, was assigned as objectively as possible, using the expectations and requirements set out in the assignment corresponding to the artifact. A more thorough description of each score and its meaning is shown in Table I

TABLE I
RATING MEANING

Value	Completeness	Quality
0	No submission	No submission
1	0-30% of features covered (At most one area addressed)	Low quality
2	30-50% of features covered (Most areas not addressed)	Adequate quality
3	50-70% of features covered (Missed a couple major areas)	Moderate quality
4	70-90% of features covered (Missed a major area)	Medium quality
5	90-100% features covered (Minor omissions only)	High quality (no major revisions needed)

C. Threats to Validity

As a reporting on the experience of using an MVP to mitigate the effects of severely shortened terms, the amount of available data does not afford a level of statistical significance. Additionally the amount of variance between the different sections makes it difficult to only control for the length of the term: projects differ in complexity and scope, mentors prioritize different parts of the design process, and clients have have different styles of interaction with the students. Since all summer length sessions had the same mentor,those results are more directly comparable. The full length term had a different mentor with more potential for variation.

While the projects were similar in complexity, the difficulty in the projects stemmed from different factors. Some projects had somewhat more complicated data models while others required the use of third party APIs and calendars. One project required non-trivial data input validation, and the students were asked to provide different heuristics to report whether the input was too self similar. A rigorous analysis of the raw data (grades) is inappropriate in this case.

The rating system in Table I was developed to help standardize and normalize around the variances in the classes. By evaluating project artifacts independently of the grades (which can be skewed by specific circumstances during the run of the course) the variations in the administration of the classes can be mitigated. Additionally, evaluating both completeness and quality separately helps normalize situations where groups sacrificed quality to address all aspects of a more complex solution.

This is an experience report; as such many of the results come from the students', mentors', and clients' perception of success. This paper attempts to address all of these different viewpoints and balance them to provide an accurate representation of the outcomes of these very different courses.

III. COLLECTED DATA

Table II lists each group, the ratings for each of the assessment metrics, and the group's project ID. Groups 1A through 1D are the cohort that completed the course in 4 weeks and had an RFP, MVP, and supporting documentation available as a reference. (None of the other groups was provided with such a prototype.) These four groups worked on the same project (though all groups were to design a separate implementation) with the same client. The next seven groups took the course over a 7 week period. During the 7 week term, four projects were distributed to the 7 groups, and each project was proposed by a different client. The final three groups each worked on a different project during the fifteen week term. This resulted in a comparison of design artifacts from eight different projects.

While there were eight different projects, the end target for all of the projects was a web application of roughly comparable complexity. That being said, the complexity was expressed in different ways (either in the number of roles in the system, or syncing with external APIs, or added requirements that created an interesting element in the project to satisfy the client need). Much of the complexity of the fifteen week projects came from shifting client expectations and needs; the projects themselves seemed to have relatively straightforward designs. This is inherently the nature of a client-driven software in a design/development capstone course. However the similarity was sufficient to allow the identification and analysis of the common elements across all of the projects and implementations.

IV. RESULTS AND DATA ANALYSIS

A. Faculty Assessments

A summary of the results by term duration is presented in Table III. The following sections summarize the analysis for each of the areas. While sample sizes are small enough that it is difficult to claim statistical significance, it is never the less interesting to compare the averages in each of the areas.

1) Data Model: Comparing the results of the data model for the shortened sections, the 7 week projects had a higher average. Some of this difference could be explained by two factors. First, the CS project had data needs that were clearly set out by the client and were restricted by the nature of the project. Second, while both of the two CW project prototypes had a data model, the documentation of those models was not clear. Some of the 4 week teams did not spend adequate time digging into the design to build off of that existing work.

The 15 week projects had the highest average. This could be explained by having additional time for iteration or a shift in mentor emphasis (the mentor for the shortened sections

TABLE II
PROJECT RATINGS

Group ID	Weeks	Data Model	UI	Requirements	Use Cases	Plan	Total	Total No Plan	Project ID
1A	4	5/3=8	5/4=9	4/4=8	4/2=6	3/2=5	21/15 = 36 (72%)	18/13 = 31 (77.5%)	CW
1B	4	4/3=7	5/4=9	5/5=10	5/4=9	4/4=8	23/21 = 44 (88%)	19/17 = 36 (90%)	CW
1C	4	3/3=6	5/4=9	3/4=7	3/2=5	3/2=5	17/14 = 31 (62%)	14/12 = 26 (65%)	CW
1D	4	3/4=7	4/3=7	5/4=9	3/3=6	3/2=5	18/16 = 34 (68%)	15/14 = 29 (72.5%)	CW
2A	7	4/3=7	0/0=0	4/3=7	4/5=9	3/3=6	15/14 = 29 (58%)	12/11 = 23 (57.5%)	DD
2B	7	3/3=6	4/3=7	4/4=8	5/3=8	4/3=7	20/16 = 36 (72%)	16/13 = 29 (72.5%)	REP
2C	7	3/4=7	4/4=8	4/3=7	5/4=9	3/2=5	19/17 = 36 (72%)	16/15 = 31 (77.5%)	REP
2D	7	5/4=9	5/4=9	5/5=10	5/4=9	4/3=7	24/20 = 44 (88%)	20/17 = 37 (92.5%)	DD
3A	7	4/5=9	3/4=7	5/3=8	5/3=8	4/4=8	21/19 = 40 (80%)	17/15 = 32 (80%)	CS
3B	7	4/3=7	3/3=6	5/4=9	5/5=10	3/2=5	20/17 = 37 (74%)	17/15 = 32 (80%)	REG
3C	7	5/5=10	4/4=8	4/4=8	5/3=8	5/4=9	23/20 = 43 (86%)	18/16 = 34 (85%)	CS
4A	15	3/4=7	2/2=4	3/4=7	3/3=6	2/2=4	13/15 = 28 (56%)	11/13 = 24 (60%)	LD1
4B	15	4/4=8	4/5=9	4/5=9	3/3=6	0/0=0	15/17 = 32 (64%)	15/17 = 32 (80%)	HWS
4C	15	5/4=9	4/5=9	4/5=9	2/2=4	1/1=2	16/17 = 33 (66%)	15/16 = 31 (77.5%)	LD2

TABLE III
COMPARISON BY COURSE LENGTH (MEANS)

Weeks	Data Model	UI	Requirements	Use Cases	Plan	Mean	Mean (No Plan)	Std. Dev (No Plan)
4	7.0	8.5	8.5	6.5	5.75	7.25	7.62	0.075
7	7.86	6.43/7.5	8.14	8.71	6.71	7.57	7.79	0.074
15	8.0	7.33	8.33	5.33	2.0/3.0	6.198	7.25	0.083

emphasized requirements and problem statements). Additionally the required data models for the 15 week section were somewhat simpler than those for the shorter sections.

2) UI Sketches: Two means are reported for the 7 week sections for UI; this reflects the mean both with and without the 0 for team 2A in this category. In this area, the 4 week projects had a significantly higher average. The teams working on CW spent a fair amount of time looking at the working prototypes and were successful in using it as starting point for their own work.

The UIs for the 15 week section were evaluated from the final presentation; these may not have been complete. Additionally in conversations with the mentor it became clear that one of the clients had shifting expectations and requirements; this was reflected in the quality and completeness of the UI Sketches.

3) Functional Requirements: While the 4 week average is slightly better than the 7 week average, the results are comparable. Again, the CW project teams were able to use the prototypes along with the RFP and supporting documentation to get high quality requirements.

The results for the 15 week sections were interesting; where the instructor of the shortened sections required the functional requirements to be formatted as "system shalls" (or similar) the instructor for the standard length section had no such requirement (or at least not as a graded artifact). The groups' understanding of the functional requirements was deduced from the final presentation and other supporting design documentation. It is interesting that the difference in the quality and completeness is not too different than the 4 and 7 week shortened sections.

4) Use Cases: The 7 week average on the use-cases was much better than that of the 4 week. While it is always difficult to ascribe reasons to differences, It is likely that two factors played a role. First, The prototypes had user stories instead of use-cases. While they are similar, there are enough differences in the required format that the quality suffered. Second, there was less time for the mentor to give feedback and have the students revise the use-cases in the compressed session.

Another possible factor is that the students during the 7 week course took more time to identify and succinctly formulate a problem statement; this additional time spent analyzing the problem from a higher level may have improved the use cases, as the student understanding of the problem itself was more thorough. This is one example of a situation where students in the longer course had a chance to explore and understand the problem space and demonstrate higher level thinking.

For the 15 week section the use cases didn't seem to go through more than one iteration. This may be due to instructor priorities and interactions with the client that cannot be controlled for.

5) Initial Iteration Plan: The 7 week group average was better, but neither session did especially well. This can be partially explained since this was the first time in the course that students were asked to write a formal plan and as a last day assignment, did not undergo a revision process. One expects that the implementation iteration plans will improve throughout the second semester of the capstone as the students achieve more practice.

All in all, the totals were comparable across the two shortened approaches. The one clear advantage for the 4 week course with prototype, was in the interface design. It follows that the combination of the RFP and MVP did mitigate for the compressed schedule, but that there were still deficits in outcomes.

The standard length section did not seem to put much of an

emphasis on the initial iteration plan; it was barely mentioned or fleshed out for any of the groups. The means including and excluding the zero score are provided.

6) Overall: When taking into consideration the scores for the planning the shortened sections seem to not show a significant difference. Many of the differences can be explained through observations of the students and the clients.

When excluding the plans from the overall mean a different story emerges. All classes reach comparable places (the mean is out of ten; these would all be considered a C in a standard grading scheme). Many of the differences can be attributed to the differences in the instructors' areas of emphasis. Some of the variance can be attributed to the required formats for submission which leads to variances in the evaluation methodology. Interestingly, even though the scores have some variance the spread of those scores remains similar for the groups of classes.

It is also instructive to look at range of totals (65% to 90%) for the 4 week groups. Since all groups worked with the same project, client, and mentor, the differences can be attributed to the students. Students contribute in several ways to the successful realization of academic outcomes, and these contributions are significantly impacted by many factors, including:

- individual strengths and weaknesses
- individual motivations and efforts
- group dynamic and interpersonal communication skills

The groups during the 4 week section displayed a wide spectrum of proficiency in these areas; the groups' performances in the second semester course supported these observations.

V. Reflections

A. Mentor Reflections

The following thoughts come from observations of the 4- and 7- week groups.

Each project had a different client, and each client brings a different level of specificity and preparedness to the process. The client who worked with groups 3A and 3C came prepared with both a very good idea of what they needed and potential UI sketches (that were not immediately shared with the students). This client had a background in design and knew exactly what processes needed to be improved through software and was easily able to demonstrate the process for the students. In comparison, other clients came in with only a vague idea in mind of what needed to be improved and worked with the students to create more firm set of requirements and use cases. The more prepared client offered an easier experience for both students and the mentor.

Along the same lines, student enthusiasm and engagement in a project can influence how much effort a student is willing to put into the learning process. The same client above who came prepared was able to demonstrate the process for the students in an exciting way. This involved "field trips" to the client's location on campus, and the students got to experience the process first hand, taking part in the client's job for the afternoon. This sparked student interest in the project, and they were noticeably more excited than some other groups were about their project. This corresponded to an easier time getting the students to spend the time outside of class integrating client and mentor feedback into their project. While this difference cannot be measured and is purely anecdotal, working with these groups was significantly easier from a mentor's point of view.

An interesting aspect discovered in the running of the 4 week version of the course is that the students who were provided a working prototype did not take full advantage of the resources provided. Groups seemed reluctant to fully explore the UI, MVP, and documentation provided to them and this became clear in the groups' discussions with the client and mentor. There was the potential for the groups to extract more information from the prototypes that was squandered, and as such the groups who were part of the shortened term had the possibility to end up further along than previous groups.

An element of the shortened course that increased the difficulty of course delivery was the depth and speed with which feedback needed to be provided to students. For the iterations to make positive progress the students needed specific feedback within 24 hours on their submissions to see where they went wrong. What made this especially challenging was the fact that the feedback could not be provided in a way that encouraged the students to explore, but instead directed them to a "correct" answer. In addition this directed feedback couldn't just be in the form of the correct answer; providing students with more specific directions to the goal without "giving away" the desired outcome was a unique challenge in this timeframe. Generating high quality feedback takes both time and effort; providing it within one day while still performing the rest of the required course administration tasks was daunting.

In summary, it is interesting to note that the project artifacts were compared for classes that ran during the summer session. In previous summers a 7 week term was allowed, though this was shortened to 4 weeks. Both of these sessions are shorter than a standard semester length (15 weeks); the biggest difference in the expected outcomes for this particular instructor was that during a full semester the class is expected to complete a mostly working prototype implementation that can be iterated on and improved during the second semester of the course. Even during the 7 week course the students were expected to implement some proof of concept portions of the project; this coding activity was not reasonable to perform during the 4 week term.

B. Client Reflections

The following thoughts come from observations of the 4 week groups.

There was an uneven incorporation/exploration of the provided documents (RFP, MVP, User Stories) by the student groups in the 4-week format. The groups that embraced using these documents made better choices. They assessed the usability of the MVP to help them explore the design space.

Ultimately, this translated into a successful implementation in the second semester. The groups that stayed at the surface level made changes in the design, but it seemed they were making changes for the sake of change. For instance, some groups added requirements that were unwarranted or made assumptions about the operation of the project that went against the RFP. One aspect that students tended to overlook in their re-imagining of the design was the requirement that certain parts of the web application needed to display well on an iPhone 5s.

One unexpected benefit of the 4-week format was that the three hour class meeting period facilitated better interactions with the client. Meeting once a week was easy to arrange. The longer meeting time allowed for an initial group discussion followed by break-out sessions with each of the groups afterwards. In a traditional semester with one hour classes, 12 meetings are required to achieve the same number of contact hours, and managing an equitable distribution of time is more difficult.

C. Student Reflections

In the last class period, the student groups in the 4 week class were asked to complete a sequence of reflection questions and then the responses were examined to get a better picture of how the students used the existing documents and what their experience was in the course. The reflection survey contained the following questions:

1) What scenarios did you test in the original prototype provided to you? How did you test them?
2) How did you apply the results of your usability scenarios to your design of the requirements?
3) How did you apply the results of your usability scenarios to your design of the data model?
4) How did you apply the results of your usability scenarios to your design of the UI?
5) Are there requirements that you could not discover using only the prototype? What are they? (Admin alone is not an answer)
6) Did your team find the user stories/use cases, functional requirements, and other supporting documentation in the repositories for the prototypes? How did you use this documentation to inform your design choices? What was reusable?
7) What pieces (if any) of the prototype do you foresee being reusable in your implementation?
8) Was the prototype self-documenting, or were there steps that were not clear how to accomplish? How do you plan to address the flaws in the prototype?
9) How did you use the formal RFP in your design process?
10) Are there requirements you could not discover using the RFP?
11) What did you learn during this process that you can apply to future design endeavors?

It was hard to draw any strong conclusions from the reflections, since the responses from most of the groups were not well thought out and lacked detail. It was clear that one of the groups did spend a significant amount of productive time working with the prototype and used it to inform their design. While there was evidence that the other three groups did use the prototype, it was unclear from the reflections how extensively they used it while creating their design.

VI. Recommendations

A. Motivations and Challenges

The compressed calendar time for the course poses a number of challenges. First, If one uses the traditional prescription requiring 2-3 hours of work outside the class period for each credit unit [19], then students should do about 24 to 36 hours of work outside of class in each of the four weeks. This does not leave much time for a student to work, take other classes or even just take a much needed break from studying.

Additionally, the software design and development process is complex and difficult for novice developers (as well as experienced veterans in the field). A course in which students must iteratively design and implement a solution to a problem requires involvement from the student beyond the lecture time.

Evaluating and understanding a problem enough to design and implement a solution is a difficult process, and can be challenging to present in a few class periods. Students require calendar time to digest and explore the problem space to understand why a good solution is good and a bad solution is not. Additionally, as the class moves forward and the students have produced more artifacts they require more time to update the documents and incorporate mentor and client feedback. This again corresponds to more calendar time. Compressing the course into a shortened time frame, while meeting the requirements for number of meeting minutes, reduces the amount of time a novice can explore and understand the problem space and solutions.

B. Recommendations for Compressed Terms

The following recommendations are offered based on the results and reflections if such a course must be offered in a shortened term:

- Use an MVP and RFP as part of the design process with an active and responsive client.
- Have the instructor serve as client to help design projects with a clear path towards success.

Reduced exploration time can be ameliorated by using an MVP and RFP as part of the design process. These artifacts provide students a more fully formed set of requirements and a working starting point for their design, enabling them to see and discuss what a ”good” solution might look like. Engaging with a working prototype provides clarification on work flow, screen content, and allows them to refine requirements based on client reactions to the prototype. Clients should be engaged, active, and ready to communicate what they like and don’t like about the work done thus far. They should be able to interact with the students and help them elicit their requirements for deletions, additions, and modifications to the current project. Access to the provided MVP data schema and other components helps students build, refine, or rework the same types

of artifacts to help them be ready to implement a complete, working solution starting the next semester. This approach reduces the time for broader exploration (potentially reducing the associated educational value), while still allowing students the opportunity to go through the requirements elicitation process and design a software system to solve a specific problem. Students are encouraged to choose to use the existing stack of technologies, or to treat the prototype as a visual specification, and are free to implement the project in the stack of their choosing. This approach helps achieve student outcomes that satisfy the requirements of the course's placement in the curriculum within a shorter time-frame course.

The second recommendation is to have the course mentor serve as the client, enabling them to help carefully design the projects with a clear path towards success. Following this approach, the mentor (as client) has a complete set of requirements clearly in mind and is often able to give more succinct and thorough feedback, helping to control spending too much time on nonproductive paths during student exploration. Using the mentor as the client can deprive students of a more realistic experience while communicating with clients, while still providing an opportunity to allow students some limited exploration in the problem space.

Following either or both of these recommendations, requires a hands-on approach from the mentor, and if not done carefully could deprive students of a complete experience of investigating a problem space, identifying solution use cases, and eliciting requirements. Particularly with the second recommendation, one must be careful to not have the mentor telling the students exactly how to get from the start to the design to the solution, and could remove some of the agency and creativity the students can bring to the problem space.

VII. Conclusions

Education is evolving, and course delivery techniques must change to meet these shifting requirements. The introduction of shortened terms and intersession classes means that instructors must adapt content delivery to be appropriate and effective within a shortened time frame. Classes involving a significant development exercise or project component can be challenging to present over a shorter term since more time allows students to more fully explore a problem space, conduct experiments, and discover solutions. It is inevitable that educational experience changes (for better or for worse) when the length of the term is quartered or even halved.

The use of an MVP to ameliorate some of the challenges that come with a loss of calendar time, and shifts the educational benefits. Students had less calendar time to more broadly explore their problem space, conduct experiments, and consider a variety of approaches, learning not only what works, but also what does not work and *why*. Providing an MVP directs the students' efforts and allows them to achieve a comparable set of outcomes over less time in an environment that is more similar to industry working schedules. Another observed effect of this condensed timeline for the class was an increased frequency of client contact. In a full length class the students may attempt to meet with the client once a week; due to faster iterations and a reduced length of obligations the students were able to meet with the client almost daily. The shortening of this feedback loop helped keep students on task and making progress.

While the previous design work can be beneficial to students there can be some pitfalls that all involved parties should be aware of. Some students seemed determined to completely redesign all parts of the provided MVPs. In some situations this was appropriate, but in others the students wanted to design away from well built and presented solutions. Students also seemed more likely to consider surface level aspects of the provided designs and were less willing to dig through code or detailed design documents to find small (sometimes important) details. The provided prototypes all had a working data model (represented as schemas in code); some groups showed an unwillingness to go and dig through the code base to find what was a well thought out and mostly successful data model).

An important takeaway for any faculty member tasked with teaching such a shortened course is that the time and effort investment is significantly higher than for a full 15 week term. Feedback on artifacts must be thorough and directed, and this deep analysis of the deliverables must be done in a condensed time frame. In addition, procuring an MVP from a previous course may not always be possible. In this case, accommodating a shorter schedule may be facilitated by removing the external client from the process and allowing the mentor to serve as client. Ensuring that the mentor/client is well-versed in the project means they can provide clear, consistent guidance and catch student missteps early. This is more removed from a real world experience, but allows the students to work through a design process in preparation for the second capstone course. Another option is that the mentor designs and implements the MVP; this can require a significant effort and time investment during course preparation, and can increase the teaching difficulty. In either case the preparation time and work load of the faculty member increases significantly.

The shortened terms still share the same challenges as the full length term which does not guarantee successful completion of the target academic outcomes. In these classes there are three primary parties: students, mentors, and clients. If any one of these is unwilling to put the time and effort into this course the resulting design and artifacts will be sub-par. A good mentor can work to manage client expectations and direct student efforts. If the students are unwilling to cooperate and put the work in (which is not unheard of) the mentor can only do so much; the resulting design will still suffer. A client who is unsure of what they want or is difficult to schedule time with will adversely affect the final product.

In summary, in longer courses, students have more time to explore, create, and discover both good and bad solutions to a problem; in shorter courses, they can be more directly guided to a positive set of outcomes that meet the needs of the particular project and provide an effective learning experience. Mentors of such classes can set the students up for success, but will need to invest more time and effort to do so.

ACKNOWLEDGMENT

Thanks to our graduate directed project clients, students, faculty, staff, and industry partners working to actively support high-quality student engagement and outcomes.

REFERENCES

[1] J. B. Thompson, H. M. Edwards, and C. J. Hardy, "Providing masters level computing students with real-life learning experiences via capstone projects," in *Information and Communication Technologies and Real-Life Learning*. Springer, 2005, pp. 231–242.

[2] A. J. Dutson, R. H. Todd, S. P. Magleby, and C. D. Sorensen, "A review of literature on teaching engineering design through project-oriented capstone courses," *Journal of Engineering Education*, vol. 86, no. 1, pp. 17–28, 1997.

[3] H. J. Ellis, G. W. Hislop, S. Jackson, and L. Postner, "Team project experiences in humanitarian free and open source software (hfoss)," *ACM Transactions on Computing Education (TOCE)*, vol. 15, no. 4, p. 18, 2015.

[4] R. E. Anderson, G. Borriello, H. Martin, and L. Black, "Capstone projects as community connectors," *Journal of Computing Sciences in Colleges*, vol. 25, no. 1, pp. 116–122, 2009.

[5] R. E. Bruhn and J. Camp, "Capstone course creates useful business products and corporate-ready students," *ACM SIGCSE Bulletin*, vol. 36, no. 2, pp. 87–92, 2004.

[6] A. Bloomfield, M. Sherriff, and K. Williams, "A service learning practicum capstone," in *Proceedings of the 45th ACM technical symposium on Computer science education*. ACM, 2014, pp. 265–270.

[7] M. Buckley, H. Kershner, K. Schindler, C. Alphonce, and J. Braswell, "Benefits of using socially-relevant projects in computer science and engineering education," in *ACM SIGCSE Bulletin*, vol. 36, no. 1. ACM, 2004, pp. 482–486.

[8] K.-B. Yue, D. De Silva, D. Kim, M. Aktepe, S. Nagle, C. Boerger, A. Jain, and S. Verma, "Building real world domain-specific social network websites as a capstone project." *Journal of Information Systems Education*, vol. 20, no. 1, 2009.

[9] Y. Kharitonova, Y. Luo, and J. Park, "Redesigning a software development course as a preparation for a capstone: An experience report," in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. ACM, 2019, pp. 153–159.

[10] C. Anslow and F. Maurer, "An experience report at teaching a group based agile software development project course," in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. ACM, 2015, pp. 500–505.

[11] J. H. Vu, N. Frojd, C. Shenkel-Therolf, and D. S. Janzen, "Evaluating test-driven development in an industry-sponsored capstone project," in *2009 Sixth International Conference on Information Technology: New Generations*. IEEE, 2009, pp. 229–234.

[12] J. Lawrance, S. Jung, and C. Wiseman, "Git on the cloud in the classroom," in *Proceeding of the 44th ACM technical symposium on Computer science education*. ACM, 2013, pp. 639–644.

[13] D. M. Case, N. W. Eloe, and J. L. Leopold, "Scaffolding version control into the computer science curriculum," in *Proceedings of the 2016 International Workshop on Distance Education Technology (in conjunction with the 22nd International Conference on Distributed Multimedia Systems (DMS'16))*. Salerno, Italy: Knowledge Systems Institute Graduate School, Nov 2016, pp. 175–183. [Online]. Available: https://doi.org/10.18293/DMS2016-036

[14] V. Isomöttönen and M. Cochez, "Challenges and confusions in learning version control with git," in *International Conference on Information and Communication Technologies in Education, Research, and Industrial Applications*. Springer, 2014, pp. 178–193.

[15] D. M. Case and C. Hoot, "Capstone as consulting," *The Journal of Computing Sciences in Colleges*, p. 67, 2019.

[16] J. Parrish Jr, J. Bailey, and B. Jensen, "Using the imagine cup sdi as the foundation for computer science capstone projects," in *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, 2010, pp. 68–71.

[17] J. D. Diaz-Mosquera, P. Sanabria, A. Neyem, D. Parra, and J. Navon, "Enriching capstone project-based learning experiences using a crowdsourcing recommender engine," in *2017 IEEE/ACM 4th International Workshop on CrowdSourcing in Software Engineering (CSI-SE)*. IEEE, 2017, pp. 25–29.

[18] A. Neyem, J. I. Benedetto, and A. F. Chacon, "Improving software engineering education through an empirical approach: lessons learned from capstone teaching experiences," in *Proceedings of the 45th ACM technical symposium on Computer science education*. ACM, 2014, pp. 391–396.

[19] J. M. Heffernan, "The credibility of the credit hour: The history, use, and shortcomings of the credit system," *The Journal of Higher Education*, vol. 44, no. 1, pp. 61–72, 1973. [Online]. Available: https://doi.org/10.1080/00221546.1973.11776844