Fostering Entrepreneurship in Project-Based Software Engineering Courses

Dr. Kevin Buffardi, California State University, Chico

Dr. Buffardi is an Associate Professor of Computer Science at California State University, Chico. After gaining industry experience as a usability and human factors engineering specialist, he earned a Ph.D. in Computer Science from Virginia Tech. His research concentrates on software engineering education, software testing, and eLearning tools.

David Rahn, California State University, Chico

Mr. Rahn is a Lecturer for Strategy and Entrepreneurship and is the Director of the e-Incubator within the Center for entrepreneurship at California State University, Chico. Mr. Rahn has extensive industry background with software and consulting startups and specialized in new product and market development. Following his successful industry career Mr. Rahn transitioned to teaching strategy and entrepreneurship at Chico State. Over the past 16 years Mr. Rahn has developed the e-Incubator at Chico State, as well as created a course called Web-based entrepreneurship which focuses on helping students launch the online portion of their businesses using the Lean Startup approach. In 2016 he published "e-Business for Entrepreneurs," an online course for entrepreneurs building e-businesses.

Fostering Entrepreneurship in Project-Based Software Engineering Courses

Introduction.

The 2013 ASEE report on Transforming Undergraduate Engineering Education identified entrepreneurship and intrapreneurship as in-demand skills that require additional attention in engineering curricula for "expanding on business and economics acumen and enabling students to learn more than economic capitalization, but also the process of starting a business from an idea" [1]. Meanwhile, the technology sector is growing, led largely by software companies like Apple, Microsoft, Alphabet (parent of Google), and Facebook. Accordingly, many of the leading software companies emerged from a "tech startup" culture and through innovations from entrepreneurial software developers.

Software Engineering courses often follow Problem-Based Learning (PBL) pedagogy by involving students in semester-long software development projects. However, previous research has identified flaws in students developing "toy" projects of their own imagination. In particular, toy projects usually lack real customers and consequently cannot benefit from their feedback. Consequently, toy projects are ineffective at holding students accountable for adapting to changing requirements – a common characteristic to real life software development, and a primary inspiration for popular Agile Software Development methods.

Meanwhile, Entrepreneurship courses are adapting to evolving processes and techniques practiced by modern enterprises. The traditional business plan approach bears similarities to the rigid planning and design of the Waterfall model and it consequently suffers from analogous inflexibility. Research has found that adopting a formal business plan is not associated with increased odds of success [2]. Instead, contemporary Lean Startup methods [3] emphasize continuous innovation through experimentation and adaptation to evolving knowledge of the market.

Accordingly, entrepreneurship curricula are evolving to provide experiential learning in adopting Lean methods. There is a movement to apply more hands-on learning in entrepreneurship education [4], including a shift toward PBL pedagogy for business venture projects [5]. While entrepreneurship students often identify software as possible solutions to market segment pain points, the entrepreneurs usually lack the development skills to see their ideas to fruition. Accordingly, there is also emerging demand for entrepreneurship PBL to embrace interdisciplinary teams that leverage specializations in different domains, including computer science [6].

In 2016, we adopted the Tech Startup model [7] of coordinating Entrepreneurship and Software Engineering classes by collaborating on novel software ideas. Unlike toy projects, Tech Startup projects leverage Entrepreneurship students to provide feedback and changing requirements as they adopt Lean Startup methods. At the beginning of each semester, students from both classes participate in ideation activities and then pitch ideas to form interdisciplinary teams and initiate software startups.

However, in the initial semesters of the Tech Startup collaborations, nearly every project idea came from the Entrepreneurs. While those teams still benefitted from collaboration and a combination of Agile and Lean methods, we wanted to encourage Software Engineering students to also actively engage in the entrepreneurial ideation.

In this study, we analyze seven semesters of a Software Engineering course (n=290)—an upper division required course within an undergraduate Computer Science program—to investigate Software Engineering students' motivations and attitudes for creating entrepreneurial ideas. We study pre-semester surveys that gather students' preferences for projects as well as their motivations for ranking their preferences.

We also examine the effects of encouraging Software Engineering students to pitch entrepreneurial ideas by prompting ideation with an introduction to emerging technology platforms: virtual and augmented reality (VR/AR) and internet of things (IoT). In this paper, we evaluate motivations for software engineers to engage in generating innovative, entrepreneurial ideas and what factors are most appealing to them when choosing from projects to work on.

Background.

With the increasing demand for software developers, many students in computing disciplines seek careers in software engineering. Accordingly, software engineering instructors seek to prepare students with professional skills and experiences with team projects that emulate those in industry. Martin acknowledged a trend in computing education where most software that students write for programming assignments "never see the light of day." Consequently, Martin argued that "toy projects" that have no real customers or use outside of the classroom are harmful [8].

Likewise, Nurkkala and Brandle assessed common gaps between common software engineering "toy projects" and real software practice, explaining:

> *A student project is just that—a project. It is not a product in any meaningful, commercial sense. Such a nonproduct escapes the scrutiny of sales, marketing, and customer relations. It also is isolated from external forces like press reviews, competing product offerings, market share, and most importantly, user scrutiny [..] The most significant gap is that student projects seldom involve a real customer.*

They acknowledge the "vanishingly small" availability of real customers willing to assume the risks inherent to a team of students working within the confines of the classroom. They also argue the inadequacy of instructors standing in as mock customers because their investment in any project is strictly pedagogical [9]. On the other hand, a real customer has needs to be met by the software and a real entrepreneurial team depends on their product's success for their professional livelihood. Those external pressures drive evolving requirements and accountability for delivering valuable, working software.

Agile Software Development (Agile) is a popular approach in industry and is consequentially taught in most software engineering courses. Among its key principles, Agile advocates customer collaboration, responding to change, and frequent interaction between businesspeople

and developers [10]. Consequently, to practice Agile in software engineering courses, students need projects that involve interaction with customers and businesspeople, who help guide the evolution of product requirements.

Problem-Based Learning (PBL) is an active learning approach—inspired by constructivist pedagogy—that fosters learning through experiencing the process of solving a problem. Studies suggest PBL is especially conducive to long-term retention and skill development [11]. Accordingly, PBL is a popular pedagogy for software engineering courses so that students gain experience to more realistic Agile software development.

Tech Startup model and project ideation.

We introduced the Tech Startup model in 2016 to heed ASEE's call to improve entrepreneurship [1] within engineering education as well as to provide opportunities for computing students to experience more realistic software engineering projects. The model involves interdisciplinary collaboration between a Software Engineering and an Entrepreneurship course, where students form small teams to create software products that meet real customers' needs.

The software engineering students practice Scrum [12]—an Agile framework that builds and reviews working software on short intervals called sprints—and entrepreneurship students adopt Lean Startup [3] methods. Each team also creates a Slicing Pie dynamic equity agreement [13] to explicitly plan for sharing the fruits of their product's financial successes as well as accounting for potential turnover.

In previous papers, we outlined the coordinated weekly schedules [14], assignments and activities [15][16], and positive outcomes for businesses that have emerged from the courses [14]. We also found that the model lead to software engineering students' more closely adhering to Agile principles than when working on other types of PBL projects [17].

Nevertheless, in preliminary semesters, we observed a need to improve software engineering student engagement in entrepreneurial aspects of the projects. At the beginning of each academic term, we introduced students to the semester-long assignment of working on interdisciplinary Tech Startup projects. We then asked students to conceptualize project ideas and pitch their proposals to their fellow students. After completing all project pitches, we surveyed students to identify the projects they were most interested in and formed teams that included at least four software engineers and at least one entrepreneur.

While all software engineering students consequently worked on tech startup projects with entrepreneurship collaborators, we noted that many software engineers were reluctant to pitch an idea. Although the software engineering course often had enrollment two to three times the size of the entrepreneurship course, nearly all pitched ideas originated from the entrepreneurship students. Anecdotally, we also observed that software engineering students sometimes perceived entrepreneurs as their bosses or superiors and that they were working for the entrepreneurs rather than collaborating with them. Consequently, in this study, we explored how to engage software engineering students more in entrepreneurial ideation.

Method.

The purpose of this study is to understand motivations behind the projects that software engineering students prefer and to investigate how to empower them to propose innovative ideas for collaborations with business students. We studied students' feedback (n=290) from seven semesters (Fall 2016 through Fall 2020) of Tech Startup collaborations between Software Engineering and Web-Based Entrepreneurship courses. The latter is an upper-division course within California State University, Chico (CSU Chico) College of Business' Entrepreneurship option. Software Engineering is also an upper-division course and is required class for a major or minor in Computer Science at CSU Chico.

Each term began with an introduction to the semester-long project structure and requirements during the first week. During the introduction, all students were encouraged to brainstorm project ideas. In a subsequent meeting with entrepreneurship and software engineering students in the same room, students gave brief pitches to explain their ideas and attract potential collaborators.

After all pitches were complete, the professors gathered students' interest in the different ideas and formed teams based predominantly on students' preferred projects. However, the professors required each team to have an appropriate number of collaborators (as based on the professors' discretion) comprised of enough interested entrepreneurs and software engineers. Professors assured students that they would stake no claim in ownership of the projects and reassured students that if they pitched an idea that becomes a team project, they were guaranteed a spot on the team.

To gather students' preferences from the Software Engineering class, we collected surveys where students shared their motivations as well as rating and ranking the proposed projects. The survey listed each project idea (along with a brief summary, when provided by the student who pitched it) for students to rate how much they were interested in working on a project.

The rating used five-point, Likert-type scale items, where one represents "Strongly Disagree" and five represents "Strongly Agree." After rating each project idea, the students also ranked their top three choices to help account for ties. Students also provided insights into what motivated their choices by indicating (with the same Likert-type scale for agreement) their (dis)agreement with the following statements:

- My project preferences are motivated by programming languages and technologies I am already familiar with
- My project preferences are motivated by new programming languages and technologies I want to learn
- My project preferences are motivated by the potential of the product making money,
- My project preferences are motivated by the opportunities for professional/career networking,
- My project preferences are motivated by the potential of the product becoming well-known and used by many people,
- My project preferences are motivated by the problem the product addresses

To account for other potential motivations, the survey asked, "Other than the reasons already given, what makes a project appealing to you?" with free-form response.

Preliminary analysis and intervention design.

After anecdotally observing software engineering students' reluctance to propose ideas, we were alarmed by the missed opportunity to engage in entrepreneurship and take agency in their education. To verify our observations, we reviewed the projects pitched in the first three semesters of adopting the Tech Startup model. Table 1 illustrates software engineers' initial reluctance to pitch project ideas, with only about 3% of software engineering students proposing a project.

Table 1. In the first three semesters of the Tech Startup model, only 3.03% of Software Engineering students proposed a project idea.

Academic Term	Software Engineering Students	Pitched ideas from Software Engineering Students	Pitches per Software Engineer
Fall 2016	60	0	0.0000
Spring 2017	32	2	0.0625
Fall 2017	40	2	0.0500
Total	**132**	**4**	**0.0303**

We reviewed literature and found suggestions that, in comparison to their entrepreneurship students, creativity from engineering students tends to focus more on the technology (the medium used to solve a problem) and making practical and incremental progress; meanwhile, their entrepreneurship counterparts are usually driven by a clearer vision of the market [18]. With the aim to motivate more software engineers to engage in creative ideation, we secured a small grant to fund equipment and resources to appeal to their inclination toward innovative technology.

A $5000 USD grant was awarded by CSU Chico. With those funds we purchased equipment to supply students with the opportunity to work on Virtual Reality (VR), Augmented Reality (AR), and Internet of Things (IoT) projects. Specifically, we purchased the following items for a total of approximately $2000:

- HTC Vive virtual reality headset and accessories
- X Glass Enterprise (development-ready wearable smart glasses, formerly marketed as "Google Glass")
- Amazon Echo Dot smart speaker
- Google Home Mini smart speaker
- Raspberry Pi 3 Model B+ microcontroller with accessories and extra sensors

With the remaining $3000, two research assistants were hired to assemble learning materials and produce a "getting started guide" with simple example applications to help students begin development for each device. Research assistants also prepared an overview of VR, AR, IoT, their unique capabilities, broad areas of problems they can address (e.g. immersive training, home automation, smart cities, etc.), as well as a list of the devices available for students' use.

Emergent technology intervention.

After purchasing the equipment and preparing materials for selective emergent technologies (VR, AR, and IoT), we supplemented our classes' introductions to the Tech Startup projects with a brief (under 10 minute) presentation of the technologies. Our primary hypothesis was that **(H1) after priming students with a presentation on emergent technologies, software engineers would be more likely to propose entrepreneurial project ideas.**

In each of the following four semesters, we followed the same process as described in the preliminary analysis: project introduction, ideation, and project pitches, followed by surveying student preferences/motivations and subsequently, team formation. In contrast to the three preliminary *control semesters* (described above), during four subsequent *intervention semesters,* the project introduction included the presentation on emergent technology and availability of the devices.

Nevertheless, for logistical reasons outside the professors' control, the course schedules were changed in the final two semesters of this study, so they no longer met at coinciding times. In response, the professors adjusted the format of the pitches so that instead of students sharing their ideas in-person, students posted their ideas on a shared message board. Although this change in delivery was not planned as part of our intervention, we considered the possibility that it also influenced students' decisions to pitch their ideas.

Software engineers typically have less experience in public speaking and presenting elevator pitches, but they also may be more comfortable with online interactions. Consequently, we hypothesized that **(H2) software engineers would be more likely to propose entrepreneurial project ideas online than in front of a class**.

To test both hypotheses (H1 and H2), we performed an analysis of variance (ANOVA) to investigate how the likelihood of software engineers to pitch project ideas—as represented by pitches per software engineer (number of pitches by software engineers divided by enrolled software engineers) for each semester—can be explained by presence or absence of the emergent technology presentation (*IV-Intervention*) or by whether the pitches were delivered online or in person (*IV-Delivery*). However, there were not any control semesters (without the intervention) where pitches were delivered online.

In the four *IV-Intervention* semesters, nearly a quarter of software engineering students pitched ideas (M=0.23, sd=0.15) while in the control semesters without an intervention, fewer than one-in-twenty software engineers pitched (M=0.04, sd=0.03) per semester. In the final two semesters with *IV-Delivery* online pitches, nearly a third of the software engineers pitched (M=0.32, sd=0.17). In the preceding five semesters (including three control and two *IV-Intervention* semesters), only seven percent of software engineers pitched per semester (M=0.07, sd=0.05).

The ANOVA revealed that the *IV-Intervention* was associated with a significant increase in likelihood of software engineers pitching their own project ideas ($F=7.59$, $p<0.05$); however,

online delivery of pitches did not meet the critical value (α=0.05) threshold for statisticalsignificance (F=5.07, p=0.087). **The statistical significance found that the null hypothesis for H1 is rejected. This result supports our hypothesis that priming software engineering students with emergent technologies would increase the likelihood that they will pitch a project idea.**

Although there were more pitches per software engineering student in semesters with online pitch delivery than those with in-person delivery, when *IV-Delivery* was analyzed in conjunction with *IV-Intervention*, its effect only approached statistical significance (p=0.085). **There is insufficient evidence to support our hypothesis (H2) that software engineers would be more likely to pitch online instead of in-person.**

Exploring motivation and project preference.

To better understand software engineers' motivations and preferences for the projects they select, we performed post-hoc analysis of their survey responses. First, we investigated whether software engineers were more likely to be interested in projects pitched by their fellow software engineers than those pitched by entrepreneurship students. We averaged students' ratings of each project (from 1 to 5, strongly disagree to strongly agree) and compared ratings between pitches from software engineers and entrepreneurs.

We hypothesized **(H3) that software engineering students would exhibit stronger interest in pitches from software engineers than those from entrepreneurs.** However, we tested the hypothesis using the Wilcoxon-Mann-Whitney test and found no significant difference (p=0.97) between interest in software engineer pitches (M=2.6, sd=0.52) and interest in entrepreneur pitches (M=2.57, sd=0.46). Consequently, H3 is not supported because there is no observable difference between interest in software engineering and entrepreneur pitches.

As exploratory analysis, we examined what motivated software engineering students' preferences for projects. We found the average Likert-type scale ratings for how strongly they agreed with the following statements (with coding abbreviation *emphasized in parentheses*):

- My project preferences are motivated by programming languages and technologies I am already familiar with (*Familiarity*);
- My project preferences are motivated by new programming languages and technologies I want to learn (*Learning*);
- My project preferences are motivated by the potential of the product making money (*Money-making*);
- My project preferences are motivated by the opportunities for professional/career networking (*Networking*);
- My project preferences are motivated by the potential of the product becoming well-known and used by many people (*Potential*);
- My project preferences are motivated by the problem the product addresses (*Problem*).

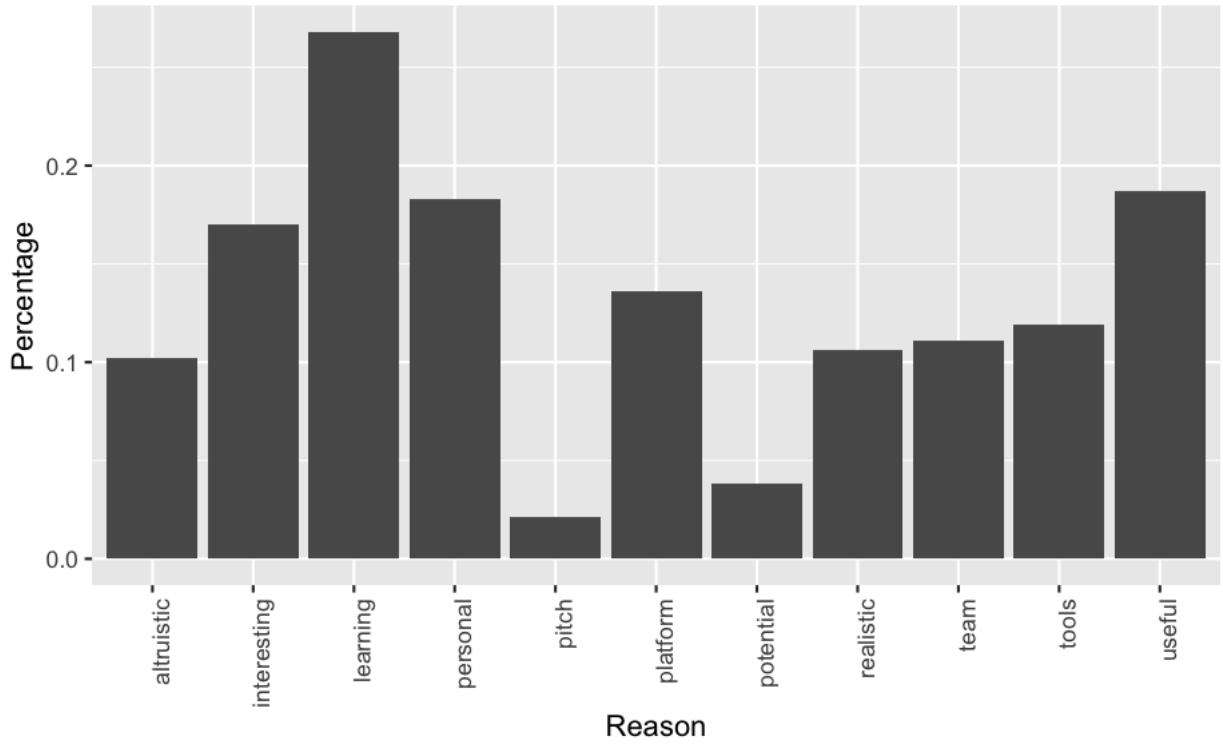
From highest (strongest agreement) to least, students rated that their project preferences were motivated by: *Learning* (M=4.06, sd=1.01), *Problem* (M=3.9, sd=1.04), *Networking* (M=3.81,

sd=1.09), *Potential* (M=3.5, sd=1.16), *Familiarity* (M=2.97, sd=1.19), and *Money-making* (M=2.62, sd=1.26).

However, we also recognize these potential motivations are not exhaustive, so we also analyzed students' free-form responses to the question, "Other than the reasons already given, what makes a project appealing to you?" Using Grounded Theory to systematically evaluate qualitative responses, we first read each response and coded them by identifying common concepts and then used inductive reasoning to categorize concepts and summarize our findings.

Each response was coded (as binary *true* or *false*) according to whether it explicitly mentioned each concept. For any given response, multiple concepts were coded when multiple were mentioned. Of the 290 students surveyed, 235 (81%) responded to the free-form response question. Figure 1 summarizes the frequency of common concepts identified and coded from the 235 responses.

Figure 1. Concept frequencies identified in free-response Grounded Theory analysis



We found the following concepts: altruism (e.g. "helping people"), interesting or challenging problem, the potential to learn, personal interest in using the product, the quality or enthusiasm in the pitch, the technology platform (e.g. mobile, web, AR, etc.), the product's potential for success, the realism of building the product, the person who pitched or other possible teammates, the specific tools involved in building the software (e.g. Python, React, etc.), and the perceived usefulness of the product. A desire to learn was the most frequently mentioned motivator.

After analyzing the concepts, we categorized motivations for project preferences according to broader themes: *infrastructure* (tools, platform), *personnel* (team, pitch), *egocentrism* (personal,

learning, interesting), *empathy* (altruistic, useful), and *outcomes* (realistic, potential). The most frequent motivations addressed how the project idea related to the survey respondent's personal (*egocentric*) goals and interests (55%). However, *empathetic* motivations (26%) of how students perceived the product affecting other people were second-most frequent. The anticipated technology infrastructure (23%) was the third most common motivation, while the project's expected outcomes (14%) and personnel (13%) were less frequent.

Discussion.

After software engineering students initially demonstrated little motivation to pitch ideas for entrepreneurial software projects, we primed ideation in following semesters with an appeal to their interest in emergent technologies. We also explored whether delivering pitches online would make software engineers more likely to pitch project ideas. Statistical analysis supported our hypothesis that priming students with a brief presentation about innovative technologies resulted in significantly more ideas pitched by software engineers. However, there was insufficient evidence to conclude whether pitching online or in-person was more conducive to engaging software engineers in entrepreneurial pitches.

Investigation of the platform for delivering pitches was not initially intended as a factor in this study. Instead, we investigated comparisons between online and in-person pitching because of unanticipated university scheduling conflicts. Consequently, the *IV-Delivery* pitch delivery treatment was only observed in tandem with the *IV-Intervention* emergent technology presentation, and not during control semesters. These circumstances were beyond our control but affected the pseudo-experimental design and represent a threat to the validity of the study.

In addition, this study only reflects the behaviors and attitudes at CSU Chico. Replication across multiple institutions would be necessary to generalize the conclusions. CSU Chico is also a recognized Hispanic Serving Institution (MSI) and enrolls disproportionately high percentage of first-generation, low-income, and under-represented minorities (URM) in STEM. Our students' motivations and behaviors may or may not reflect those of the general population of software engineering students.

Nevertheless, it is encouraging to observe the increased engagement of software engineers participating in entrepreneurial ideation. According to their self -reported motivations, software engineers were motivated the most by how each project appealed to their personal interests—particularly with a desire to learn new programming languages, frameworks, and technologies.

One might expect some students to prefer projects that will require the least effort to achieve their desired grade. However, none of the students explicitly stated in their free response answers that their project preferences were influenced by which ones they perceived to be the easiest. On one hand, that sentiment may be reflected implicitly in responses that claimed preference for projects that seemed "realistic" or "practical" to implement. On the other hand, those responses may not indicate preference for easy projects since students also reported preferences for projects that involved a technology that they would like to learn over those using familiar technology.

Preferring a project that requires learning additional skills is not consistent with the notion that students may just prefer the least amount of effort.

Moreover, the hypothetical appeal of an easy project may have been mitigated by the project requirements. All project pitches were vetted by the professors to ensure the subject matter was appropriate and of reasonable scope. The usefulness of the software product and how well a team adopted Agile methods to continuously deliver minimal viable products (MVP) and adapt to changing requirements accounted for 30% of the project's grade. The adage "Software is never finished, only abandoned" is reiterated in class to emphasize that their software needs to be continuously maintained and improved and to discourage students from treating the projects like traditional academic programming projects that have rigid requirements and can be considered "finished."

It is also worth noting that we have observed an increased variety in platforms the software projects are built upon. Before providing the emergent technology intervention, nearly all ideas involved web and mobile phone applications. Since introducing students with an overview of virtual reality (VR), augmented reality (AR), and Internet of Things (IoT) and supplying the necessary equipment, we have seen an increase in pitches involving those technologies.

As post-hoc analysis, we reviewed the technology platforms for each idea that garnered enough interest and formed a team. For this analysis, we grouped web and mobile platforms together because projects often use both—such as mobile applications that accesses a web application programming interface to maintain persistence and social networking features—and because cross-platform frameworks (e.g. Apache Cordova, Google Flutter, etc.) blur distinctions between web and mobile development. In three semesters preceding the intervention, only one project (4%) used VR while the rest (96%) used web/mobile platforms. Since introducing the intervention, web/mobile projects are still most popular (81%) but there has been an increase in VR/AR (5%) and IoT (14%).

However, we also acknowledge that these technologies have simultaneously become more pervasive in society so we cannot attribute the evolving trends in technology platforms to our intervention alone. Furthermore, return on investment should be weighed when considering the impact of priming software engineering ideation with emergent technology. While VR/AR technology piques many students' interests, the VR/AR devices we purchased accounted for most of the $2000 USD budget. Only two projects (in four semesters) have used the HTC Vive VR equipment and while several pitched projects involved AR, none of those projects used the X Glass wearable technology.

On the other hand, the remaining devices (two smart home devices and a microcontroller with addons) each cost less than $100. Several projects have used Raspberry Pi microcontrollers and while some have used the equipment we purchased, many teams opt to purchase their own equipment, in part due to its low cost. Although our intervention included both (A) a presentation to prime students to consider ideas that involve emergent technology and (B) making equipment available to students, our observations suggest that the presentation (A) probably made the greater impact on students' likelihood to pitch their own ideas.

Future Work.

Engaging software engineering students in the process of building a viable product from a novel idea shows potential for learning Agile methods while engaging in entrepreneurship. This study explored software engineering students' motivations and revealed that appealing to their interest in innovative technologies should encourage them to take agency in proposing interdisciplinary, entrepreneurial projects. We also anecdotally observed a transition of students increasingly embracing a perspective of team collaboration rather than a hierarchical client/contractor relationship.

In part, students' collaborative attitudes may be due to the required Slicing Pie dynamic equity agreement assignment [13][15] that defines each team member's role in contributing to the project outcome (along with equity earned, accordingly). Nevertheless, there remains room for improving how well students from either discipline appreciates the skills, methods, and value of their counterparts. Improving inclusivity and engagement of software engineering students in the ideation process is important but is likely only one of many factors that is vital to encouraging a collaborative team structure.

However, there is a lack of research on software engineers' self-efficacy and agency as it applies to creating novel software projects. Future research should explore self-efficacy of engineers when either engaging in interdisciplinary projects with business students or venturing into tech startups without business partners. The quasi-experimental was not initially designed for investigating how project ideas would be pitched (online versus in person) but still found differences approaching statistical significance. Consequently, future studies should include controlled experiments to understand the potential effects of the medium used for ideation and team formation.

The research outlined in this paper focuses on student agency and motivation, which should be extended with longitudinal studies of project quality as well as continuity and success of entrepreneurial projects. In previous work [14][16] we found that some teams continue to work on their project after the academic term is complete and a few even legally incorporate and launch enterprises. However, it may also be worth investigating whether the emergent technology intervention or other factors in the ideation and team formation process influence project outcomes.

Instructors should embrace software engineers' prevailing interest in learning new technologies as a motivating factor in software engineering courses. However, we should also acknowledge that the penchant for making decisions based on technology over the usefulness or viability of the product may have ill effects on project outcomes. Choosing a tool to solve a problem due to personal motivations rather than its suitability is a bad practice—it resembles the "Silver Bullet" or "Golden Hammer" software development antipattern [19]. Future work will investigate the technologies chosen for projects and their impacts on design, maintenance, and entrepreneurial success.

While the technology sector continues its rapid growth, both new technologies and new business models emerge. Historically, personal computers and desktop applications gave way to online retailers, followed by social media and services that often leverage data gathering and advertising for revenue. Subsequently, novel business models for the technology sector have emerged, spurring technological and business innovations such as the "Gig Economy" (e.g. Airbnb, Uber, etc.). Educators should consider the potential for fostering innovations in business and software in tandem with the collaborative, interdisciplinary Tech Startup model.

Conclusions.

The Tech Startup model provides an opportunity to involve software engineering students in realistic, problem-based learning while gaining novel and valuable exposure to the entrepreneurial "process of starting a business from an idea" [1]. However, in preliminary semesters, we found that only 3% of software engineering students took the initiative to propose their creative ideas among their peers and business students. Consequently, we provided equipment and a brief overview of emergent technologies to prime software engineers' entrepreneurial ideation.

In a mixed methods study (n=290), we compared the likelihood of software engineers pitching their own entrepreneurship ideas after being introduced to three emergent technologies: virtual reality, augmented reality, and internet of things. We found the software engineers' pitches increased ($p<0.05$) more than sevenfold in semesters when the emergent technology intervention (23%) was applied. We also observed project ideation with pitches delivered either in-person or via an online message board. However, further investigation is necessary to determine whether the method used to deliver pitches affect software engineer engagement.

The study also employed quantitative and qualitative analysis of students' project preference ratings and free-form responses to a survey. When identifying the projects that they prefer, software engineering students showed no preference between ideas pitched by fellow software engineers and those pitched by entrepreneurs. Software Engineering students identified that a desire to learn a new programming language, framework, or technology was the most common motivation for project selection. Students also reported motivation from how interesting they find the problem and whether they would personally use it.

References.

[1] American Society for Engineering Education, "Transforming Undergraduate Engineering Education, Phase I: Synthesizing and Integrating Industry Perspectives," Workshop Report, May 2013.

[2] A. Bhide, A. How Entrepreneurs Craft Strategies That Work. Harvard Business Review. [Online] Available: http://hbr.org/1994/03/how-entrepreneurs-craft-strategies-that-work/ar/1. [Accessed February 19th, 2014].

[3] E. Ries, The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business, 2011.

[4] E.A. Rasmussen and R. Sorheim. Action-based entrepreneurship education. Technovation, 26(2), 185-194, 2006.

[5] C. Hixson, M. Paretti, J. Lesko, and L. McNair. Course Development and Sequencing for Interdisciplinary Entrepreneurship Education. Presentation given at OPEN 2013: NCIIA's 17th Annual Conference, March 22-23, 2013, Washington, DC.

[6] P. Lane, J. Hunt, and J. Farris. Innovative teaching to engage and challenge twenty-first century entrepreneurship students: An interdisciplinary approach. Journal of Entrepreneurship Education, 14, 105-123, 2011.

[7] C. Robb, D. Rahn, and K. Buffardi. Tech startups: A model for realistic entrepreneurship & software engineering project collaboration. United States Association for Small Business and Entrepreneurship. Conference Proceedings; Boca Raton: 1280-1294. Boca Raton: United States Association for Small Business and Entrepreneurship, 2017.

[8] F. Martin, "Toy projects considered harmful." Commun. ACM 49(7), July 2006. pp. 113-116. doi:10.1145/1139922.1139958

[9] T. Nurkkala and S. Brandle, "Software studio: teaching professional software engineering." In Proceedings of the 42nd ACM technical symposium on Computer science education (SIGCSE '11). ACM, New York, NY, USA, 2011. pp. 153-158. doi:10.1145/1953163.1953209

[10] M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R.C. Martin, K. Schwaber, J. Sutherland, and D. Thomas. "Manifesto for Agile Software Development." Agile Manifesto, February 2001. [Online]. Available: Agile Alliance, http://agilemanifesto.org/.

[11] J. Strobel & A. van Barneveld. When is PBL More Effective? A Metasynthesis of Meta-analyses Comparing PBL to Conventional Classrooms. Interdisciplinary Journal of Problem-Based Learning, 3(1), 2009. doi:10.7771/1541-5015.1046

[12] J. Sutherland. "Agile Development: Lessons learned from the first Scrum" Scrum Alliance, 2004.

[13] M Moyer. "Slicing Pie: Fund Your Company Without Funds" Lake Forest, IL: Lake Shark Ventures LLC, 2012.

[14] C. Robb, D. Rahn, and K. Buffardi. Bridging the gap: A model for interdisciplinary collaboration between entrepreneurship and software engineering students. Journal of Education for Business, Routledge. August 12, 2019. pp 1-10. doi:https://doi.org/10.1080/08832323.2019.1644275

[15] K. Buffardi, "Tech Startup Learning Activities: A Formative Evaluation," 2018 IEEE/ACM International Workshop on Software Engineering Education for Millennials (SEEM), Gothenburg, 2018, pp. 24-31.

[16] K. Buffardi, W. Zamora, C. Robb, and D. Rahn. Implementing the Tech Startup Model: A Retrospective on Year One. American Society for Engineering Education (ASEE) Annual Conference & Exposition, June 23, 2018.

[17] K. Buffardi, C. Robb, and D. Rahn. Learning Agile with Tech Startup Software Engineering Projects. In Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '17). Association for Computing Machinery, New York, NY, USA, 28–33, 2017. doi:https://doi.org/10.1145/3059009.3059063

[18] H. Berglund, and K. Wennberg. Creativity among entrepreneurship students: Comparing engineering and business education. International Journal of Continuing Engineering Education and Life-Long Learning, 16(5), 366–379, 2006. doi:10.1504/IJCEELL.2006.010959

[19] P.A. Laplante and C.J. Neill. Antipatterns: identification, refactoring, and management. Auerbach Publications, New York, NY, USA, 2005.