Developing an Agile Mindset in Software Engineering Students

Suddhasvatta Das
Dr. Kevin A Gary, Arizona State University

Dr. Gary is an Associate Professor in the School of Computing and Augmented Intelligence at the Ira A. Fulton Schools of Engineering at Arizona State University. His research interests are in agile and open-source software, software engineering in healthcare, and software engineering education. Presently he is focused on flow and quality metrics derived from agile research and applied to open-source software, and in identifying Regression Test Selection methods suitable for Agile and Lean software development.

He was a founding faculty member of the software engineering degree programs at ASU and developed the project-centric curricular implementation known as the Software Enterprise. He has served twice as program chair and led the program through multiple positive ABET accreditation visits. Kevin blends industry and academic experience to bring theoretically grounded, practice-oriented methods to the classroom.

Kevin is a member of ASEE, ACM, and IEEE.

Developing an Agile Mindset in Software Engineering Students

Abstract

The agile mindset is a set of values and principles extracted from the Agile Manifesto focused on trust, responsibility, ownership, continuous improvement, and continuous openness to learning and growing. A growing body of literature and anecdotal experiences in the gray literature suggest it is a necessary component for successful agile transformations in organizations. However, many organizations are unable to reap the full benefits of agile software engineering as they focus on agile practices only. Likewise, we postulate that the fully prepared graduate from a university software engineering program should exhibit an agile mindset, instead of merely gaining competence in agile practices. In this study, we investigate the agile mindset of university students in software engineering and working in an agile project environment as a part of a course. Two courses utilized a typical approach, teaching agile development competencies in a project-centric course. A third course extended this form of learning with additional critical inquiry activities to elevate internalization of agile principles and develop an agile mindset. A custom survey was employed and analyzed the results using standard descriptive and inferential statistics to investigate the outcomes.

Introduction

Working in an agile setting demands more than just pure software engineering skills. These skills are non-technical and social, often related to communication, collaboration, and understanding of the broader goal of the business [1][2]. These skills are often scarce among software engineers, and that is why roles such as Scrum master mentors and coaches are present to develop a deep understanding of the foundations of agile [1][3]. An agile mindset suggests mastering these skills for practitioners to have an effective teamwork environment [3]. Thus, developing a proper agile mindset enhances agile project success [4]. Early career software engineers typically receive their first exposure to agile principles and methods in a university setting. Project-centric courses such as capstone experiences often expose students to the Agile Manifesto and to the mechanisms of industry-relevant agile practices. But to what extent does this exposure help these future professionals develop the agile mindset required to be successful in the modern software development organization?

Popular agile and lean methods [6] such as Scrum and Kanban are formulated on a conceptual groundwork assuming an agile mindset. More recently, the industry is scaling projects up in size and scaling agile practices out across the organization. The growing adoption of the Scaled Agile Framework (SAFe, [7]) is centered around a principle of alignment, necessitating organizational levels, from the engineer to corporate-level strategists to think agile. This industry evolution of agile methods since the inception of the Agile Manifesto [8] requires an in-depth understanding of the agile mindset, evolving toward the concept of “being and working agile” [9]. Mordi and Schoop [10] offer the most explicit definition of the agile mindset in the literature:

> *Agile Mindset is a mindset based on the values and principles of the Agile Manifesto, whose main characteristics are trust, responsibility and ownership, continuous improvement, a willingness to learn, openness, and a willingness to continually adapt and grow. It is underpinned by specific personal attributes on the individual level and an enabling environment on the organizational level, which allows autonomy of people and teams, managing uncertainty and a focus on customer value, with the goal of achieving a state of being agile instead of merely doing agile.*

Software engineering organizations hiring new university graduates look for an agile mindset in students. Thus, universities should not only focus courses on agile machinery and practices but also try developing the agile mindset of students to make them successful in the software engineering profession. Many degree programs at universities have tried to respond to this by offering new courses or updating existing ones on software processes to include agile principles and tools. Likewise, modern software engineering textbooks now include some content on Scrum and agile practices. The research community offers several papers on the understanding of teaching agile content and projects in the classroom. However, these studies focus on techniques of agile methodologies, usually Scrum and XP (scrum boards, burndown charts, test-driven development, pair programming, continuous testing) and agile practices (sprints, daily standups, retrospectives). The learning outcome for students results in knowing the machinery of how agile works, but not the agile mindset.

Future software engineers aspiring to work for software organizations that follow agile methodologies must start developing an agile mindset while in school. Companies favor hiring software engineers who know the mechanics of executing an agile process but also have the mindset to work in an agile setting. These new hires should be able to learn and be proficient in the organization's agile tools and machinery, with the mindset to execute and deliver an agile project successfully. We hypothesize that the way agile is presently taught in classrooms at best minimally achieves this goal. Thus, investigating the development of the agile mindset in students and evaluating how the current teaching methods affect the same is an important research avenue for the community to consider.

This paper presents a survey study of three university courses, two graduate and one undergraduate, that teach agile methods in a project-based setting. One graduate course extends the project-centric approach by adding critical inquiry activities to force the students to consider agile foundations more deeply. We believe this is only a first step towards reorienting university curricula around agile software engineering in a way that better prepares students for how agile has continued to evolve in practice.

Background

Software engineering programs or software engineering courses within computer science programs have responded to the popularity of agile software development (ASD) in professional practice by updating courses and capstone project offerings to utilize agile. However, course materials primarily focus on the mechanics of ASD, particularly Scrum or XP. We could only identify a few studies that focused on developing an agile mindset and teaching ASD mechanics.

Devedžić [11] presents multiple cases of agile teaching at a university. The paper claims teaching agile to university students is most effective when done in an agile way. Students learning agile should be encouraged to interact and get more involved rather than following textbook descriptions. This method, in turn, will train them to see the bigger picture and understand the small nuances of ASD. The study recommends that the role of the teacher should be of a guide on the side but not guiding them at every step.

Hedin et al. [12] present a case study on how Extreme Programming (XP) can be put into the software engineering curriculum. XP has a positive effect on the students, learning problem-solving, testing, iterations, handling customers, requirement analysis, release process, and other engineering practices fit for a midsized organization. Inexperienced students received help with architectural work and segmentation of work, so there are more iterations for them to learn agile rather than someone directing them on what to do. Pedagogically, coaching was employed, with intensive mentoring sessions to develop production-level code weekly for 8 hours.

Kropp, and Meier [13] ask three essential questions, 'Why is there a lack of appropriately skilled personnel who can deliver, 'What is the reason behind this, and how can it be improved? The results show that agile values can be taught if a proper teaching model is followed. The authors propose a teaching methodology that focuses on internalizing agile values and practices for new graduates. This new method has two parts, applying engineering practices and applying management practices. Agile values were taught in part one, and students were asked to apply those principles no later than the project's second iteration. Students were encouraged to follow both engineering and management practices throughout their projects. Agile values were regularly discussed during lectures to emphasize their importance.

Schroeder, et al. [14] answers two critical questions related to the study of the agile mindset: 'How does teaching agile impact students to have better self-organizing teams? How to deal with team members with different levels of motivation and skill levels? The authors propose ways to teach Scrum methodology to future engineers going into the profession. Suggestions on process organization as a critical factor for successful agile projects are emphasized, as this helps students measure velocity and manage resources.

Kropp et al. [15] postulate that the more experience a person has in agile software development, the better the collaboration, and successful agile teams in the industry are very efficient in collaborating. The paper shows how collaboration can be taught in a classroom setting. According to the study, agile competency can be divided into three phases: Technical Competency, Collaboration Practices, and Agile Values. After the first year, students observed improved technical competency. In the second year, agile collaboration lectures connect theoretical knowledge to actual practices.

Scharf and Koch [16] discuss the importance of simulating real-world agile projects with Scrum or XP. The authors designed an undergraduate software engineering course to provide students with a deep understanding of agile methodologies. Feedback from the students after the course was that the workload was high (> 10 hrs/wk) but acceptable. Students also preferred to work in teams and spent 2/3rd of their working time with teammates. However, the authors also report drawbacks, such as difficulty running a high-demand course, especially communication issues

with a large course staff team. Random team formation and open-source Scrum tools were also not well received by the students.

Reichlmayr [17] reports experiences running a semester-long project using ASD practices for a lower division course. Agile methods proved beneficial developing user stories that helped planning and documentation, delivering in increments that enabled them to improve and continuously improve, and enhancing teams to work collaboratively. The authors conclude that agile execution methods can also be used for upper-division courses.

Rico and Sayani [18] present a study of a master's capstone project using agile methods. The project teams had little to no experience and training, but they successfully completed the project. Results showed that teams with high velocity had the least customer satisfaction and vice-versa. Teams that complained the most needed the most supervision, relying heavily on the customer to lay down requirements, and customers with experience with agile acted as a critical success factor.

These papers are a sampling of a wider body of university teaching experiences related to ASD across software engineering and computer science programs, and from the lower division up through graduate work. We found in our review of the literature that teaching primarily focuses on ASD process machinery (Scrum ceremonies) or developer best practices (derived from XP), but indirectly on higher level values they hoped students should acquire from the experiences. To confirm these observations, we conducted a study in 3 project-centric courses in our undergraduate and graduate programs that all use ASD on projects to determine if teaching agile machinery leads to the agile mindset.

At Arizona State University, a project-centric curricular design [19] known as the Software Enterprise [20] incorporates agile methods (primarily agile machinery), while an elective graduate course in Software Agility promotes critical inquiry for deeper understanding between theoretical foundations of agile and professional practice. Our hypothesis, based on our experiences and on our review of the literature, is that teaching agile machinery does not lead to a greater agile mindset, and instead the agile mindset needs to be explicitly taught using a pedagogy supporting deeper understanding (e.g. critical inquiry).

Methodology

We employed a custom survey instrument in 3 project-centric courses to answer the following research questions:

> *RQ1: Does learning agile "machinery" lead to the development of an Agile Mindset?*
> *RQ2: Does a student's prior experience with agile methods impact the outcome of RQ1?*

This study was conducted with IRB review and approval by Arizona State University.

Survey Design. The survey had four sections: Consent, Background or Demographic Information, Recollection of Agile Concepts and Competence with Agile Machinery and Agile Mindset. Students were awarded a small amount of extra credit to complete the survey, and offered an alternate extra credit exercise if they chose at any time to opt-out of the survey. As there was no

validated instrument from the literature at the time, questions were carefully designed by the first author and reviewed by the second author based on the authors' experience in agile classrooms. The demographic questions were a part of the questionnaire to extract different subsets (s) of the population to answer RQ2, but did not reveal personally identifying information (PII).

The questions for the Recollection of Agile Concepts and Competence with Agile Machinery (later reported in this paper as the Tools score) and the Agile Mindset (later reported as the AM score) sections are given below in Table 1. All questions were multiple select questions, and most (all but questions 1 and 6) were scored on a weighted "number right" basis and scaled to 5 points each. The AVG column gives the weighted average of scores across all students taking the surveys.

Table 1. Survey Questions, Answer Choices, and Average Scores

QUESTION with ANSWER CHOICES below	**AVG**
Which of the following best describes the use of sprint burndown charts?	3.65
Which of these following should be the outcome(s) of sprint zero?	1.70
What items from the list below are relevant to a Product Backlog?	3.40
Which of the following should be in a User Story?	4.07
Which of the following are the attribute(s) of Agile Mindset?	3.68
Which of the following best defines Agile Mindset?	4.20
Which of following are not attribute(s) of Agile Mindset?	3.10
Imagine a scenario where a bug has been in production by customer. What should be the plan of action for the team?	3.52

Item scores were reasonably consistent though Question 2 on sprint zero is noticeably lower than the others. A closer inspection of the scores for each of the classes shows there is a significant difference between the best performance (SER316) and the other 2 classes, most likely explained by the emphasis on sprint zero in the teaching materials for SER316.

Target Population. We used three classes offered at Arizona State University under the Software Engineering Bachelor's and Master's programs: SER316 (undergraduate), SER515 (graduate core) and SER516 (graduate elective). This gave diverse data that included both undergraduate and graduate students with 0-5+ years of outside experience in ASD. All classes used Scrum in an extended (4 or more sprints) team project. The classes were divided into teams of 3-5 students and worked in 1-to-3-week sprints. Each team delivered working software by the end of every sprint and the end of the semester. Each team was graded on how effectively they executed agile (followed Scrum ceremonies, worked in a regular cadence, and produced working software).

SER516 is titled Software Agility, in this course students do 2 projects, one a standard Scrum and a second larger project using Scrum-of-Scrums (SoS) or Kanban. They are required to do critical inquiry activities, reading relevant papers from academic and gray literature and discussing them in class for participation points or answering essay questions on take-home assignments and quizzes.

Data Collection. Data was collected over two semesters, Fall 2021 for SER316 and SER515, and Spring 2022 for SER516. Surveys were given in the last week of class of each semester. The instructors of the three classes announced the purpose of the survey, read a briefing statement from the researchers, and delivered the survey to the students via the course learning

management system Canvas. The first question on the survey asked for informed consent, and at any point during the survey the participant could opt-out. Instructors of the courses had knowledge of which students completed the survey to award extra credit, but then exported deidentified data and provided it to the researchers, so from the researchers' standpoint the data collection was anonymous.

Data Analysis. Collected data was exported to Excel and cleaned. The combined number of responses from the undergraduate and graduate classes was 228: SER515 n=148, SER516 n=34, and SER316 n=46). We manually removed one partially filled survey from SER515, giving a final number of 227 accepted responses. Next, a grading rubric was applied to the questions in the categories of Agile Concepts and Competence with Agile Machinery and Agile Mindset . Each question was worth five points if it was correct with partial points awarded based on the grading rubric designed by the first researcher and reviewed by the second researcher. There were four questions in each section Agile Concepts and Competence with Agile Machinery and Agile Mindset in multiple choice, multiple answer format. The scores for the former category are labeled 'Tools score' and the scores for the latter category labeled as 'AM score'. The maximum and minimum points students can score on each are 20 and 0.

Results and Analysis

Considering our first research question:

RQ1: Does learning agile "machinery" lead to the development of an Agile Mindset?

To investigate RQ1, we analyzed the relationship between the Tools score (agile machinery), and the agile mindset (AM) score, representing internalization of agile values and principles.

Table 2 shows the correlation coefficients (r) of Tools and AM scores for all populations and strata. The population is stratified in two ways: 1) by class, and 2) based on whether the student had prior working experience with agile methods (including academic and/or industry experience). We classified students as *non-experienced* if they answered 'No Experience' or '0-6' months to the experience-related demographic question. All other experience levels in months from '6-12' or more were classified as *experienced*. These subsets are mutually exclusive (stratified) from the total number of responses (227) we received. We wanted to investigate this to ensure there were no combinatory effects from sampling both undergraduate and graduate students and students who may have had a fair amount of industry exposure to agile methods before taking the respective class.

Table 2. Tools/Mindset correlation by (sub)population

Population	**Number of Data Points**	**r**
Entire Population	277	0.086
SER316	46	0.077
SER515	147	0.038
SER516	34	0.444
Non-experienced	85	0.060
Experienced	142	0.094

Figure 1 shows a scatter plot of the full data for the Tools and AM scores. We plot a best-fit line to ascertain if there is any predictive power between the variables. We repeat this process for the *Experienced* and *Non-experienced* subpopulations in Figures 2 and 3, and for the 3 classes in Figures 4, 5, and 6. Table 2 and Figures 1-6 support the conclusion that Tools and AM performance are not related. We do note that in SER516 the relationship is stronger (0.444) and that some predictive power exists (0.4872), which we attribute to the overlap of teaching agile practices and agile theory at the same time during the semester.

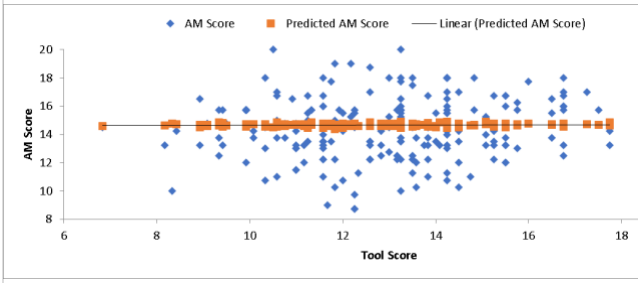

Figure 1. Scatter plot for entire population (n=277)

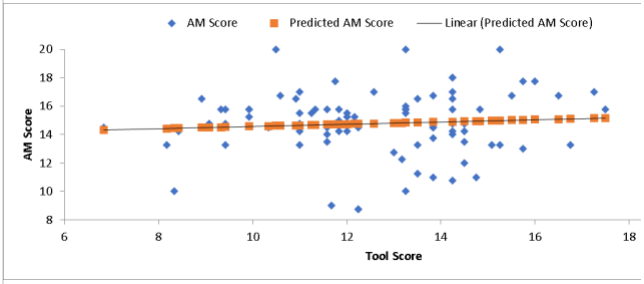

Figure 2. Scatter plot all non-experienced students (n=85)

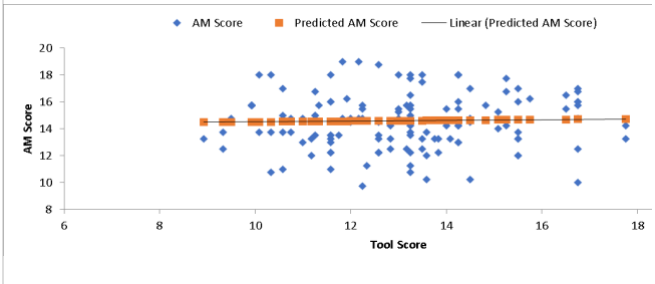

Figure 3. Scatter plot for all experienced students (n=142)

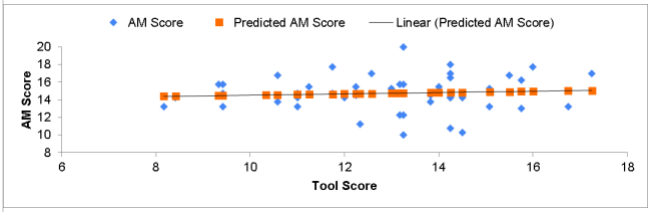

Figure 4. Scatter plot for all SER316 students (n=46)

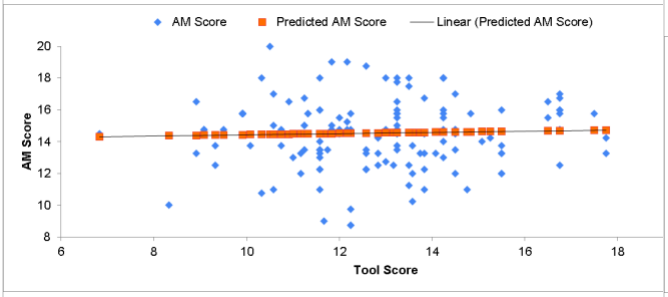

Figure 5. Scatter plot for all SER515 students (n=147)

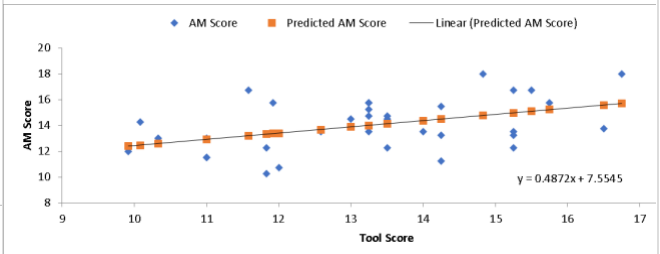

Figure 6. Scatter plot for all SER516 students (n=34)

Based on our analysis, we cannot conclude that learning the machinery of agile will lead to an agile mindset.

RQ2: Does a student's prior experience with agile methods impact the outcome of RQ1?

To answer this question, we stratified the data once by class, and a second time by experience and re-analyzed the same way for these mutually exclusive subpopulations. Looking first at the courses, we used descriptive statistics to summarize the data distribution for the three classes. We computed the correlation between the AM score and Tools score for the entire population

and then stratified the data by the 3 classes and then differently by experience. We felt these mutually exclusive strata would provide us sufficient insight as to whether year in school or years of experience impact our hypothesis outcome. We used scatter plots to visualize the data for spread and any rate of change (trend) information.

Table 3 shows the descriptive statistics for the distribution of Tools and AM scores for the 3 classes, while Figure 7 displays box-and-whisker plots for this data. Note that although the AM scores are higher in all 3 classes compared to the respective Tools score, there is no basis for direct comparison as they are distinction survey items.

Table 3. Tools and Agile Mindset Scores by Course

	SER316 (46)		**SER515 (147)**		**SER516 (34)**	
	Tools	*AM*	*Tools*	*AM*	*Tools*	*AM*
Max	17.25	20	17.75	20	16.75	18
Range	9.08	10	10.92	11.25	6.83	7.75
Mean	12.84	14.73	12.71	14.54	13.29	14.03
Median	13.21	14.63	12.58	14.75	13.25	13.63
StdDev	2.22	2.07	2.14	2.17	1.84	2.02

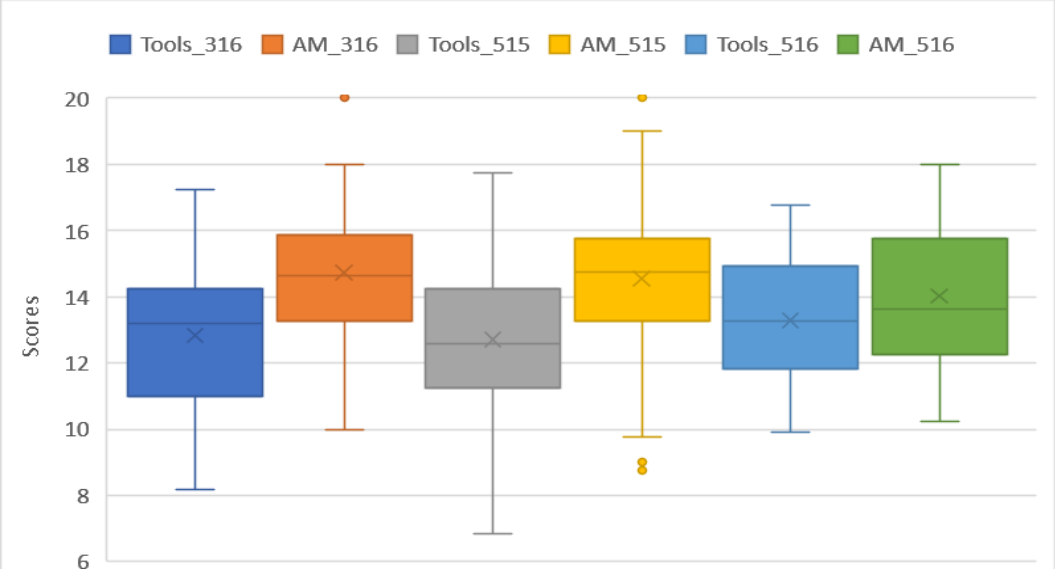

Figure 7. Tools and Agile Mindset score distributions for the 3 courses

This table and figure show there is no significant difference between classes in performance on either survey, and unpaired t-tests confirm this (316-515 Tools, AM: p=0.743, 0.613; 515-516 Tools, AM: p=0.141, 0.204; 316-516 Tools, AM: p=0.324, 0.133).

One observation is that the undergraduate course students (SER316) performed better than graduate students on both components of the survey. The Tools component can be explained by differences in the sprint zero question (see Table I). A second observation is that SER516 students scored the lowest of the 3 classes on the AM component which runs counter to the one of the objectives of the course.

Considering the experienced versus non-experienced stratum next, we note there really is not any significant difference in the relationship between machinery and mindset (Table I, Figures 2 and 3), and little difference in the trend line as well. Therefore, consistent with our other results, exploring this subpopulation also does not lead to any significant relationship between machinery and mindset. It is possible that our partition of 0-6 months of industry experience did not recognize partial experience (the overlap between industry and academic experience, or students who have one or more internships and may get some exposure to mature agile processes that way) properly, as we thought this would have yielded some observable difference.

This analysis essentially shows there is no significant relationship between learning agile practices and developing an agile mindset. This aligns with our hypothesis, presented earlier, that teaching agile machinery does not lead to a greater agile mindset.

Threats to Validity

Internal Validity: This threat concerns parameters that affect the relationship between the research activity and the results, such as participant bias. We mitigated this issue by selecting classes that use ASD practices for a semester-long project to avoid unreliable data. Another threat not controlled for is instructor bias, and finally we note there were changes in classroom restrictions due Covid-19 between the Fall and Spring semesters in the 2021-22 academic year.

External Validity: This threat is concerned with the generalizability of our findings. To address this, we had participants ranging from zero to sixty months of experience with a background in ASD. We also analyzed both an undergraduate class and two graduate classes. We note that a vast majority of our graduate population is made up of international students with international work experience and a different undergraduate academic setting.

Construct Validity: This threat concerns whether the questions in the survey represent the attributes under measurement. We carefully designed the survey, but we had to limit the length of the survey to 20 minutes, shorter than we wished and possibly and missed attributes for measurement. Unfortunately, we could not mitigate this issue, which is a prevalent problem in survey research.

Conclusion Validity: This concerns the ability to draw proper conclusions. The survey is a custom instrument, as there is no existing instrument for the agile mindset in the literature. A new 20-question instrument was published [21] after this study, but this instrument is only suitable for industry teams and organizations, not for a university setting. We also note there may be alternative interpretations to why SER516 show a stronger relationship between survey component scores; for example, it could be due to SER516 being taken (typically) after SER515 in the graduate curriculum.

One additional limitation is we may have students who randomly selected options without paying attention to the questions to earn extra credit for finishing the survey. We attempted to mitigate this by providing an alternate extra credit activity in each course. Finally, the survey instrument was developed and scored by a PhD student though reviewed by a senior professor. Inconsistent design and scoring may cloud the data.

Conclusions and Future Work

The next generation of agile is underway, as organizations work to transform to large-scale agile adoption [22]. Organizations are realizing that to adapt agile to scale, the organization must adopt an agile mindset [21]. The most efficient way for organizations to do this is to develop an agile mindset in its workforce, including seeking to hire new college graduates that know more than the mechanisms of agile processes but internalize agile values and principles.

This initial work suggests that teaching agile practices, common now in university curricula, does not lead to an agile mindset, and that more is required to develop the mindset in students. It is an open question as to the best way to do this, and to what extent a student needs to demonstrate the mindset compared to what is expected of professionally mature software engineers. In our work we are continuing to refine our pedagogy for teaching agile. The SER516 class emphasizes critical inquiry through paper reading, discussions, and personal reflection through free response assignment questions and concept mapping. However, in these initial results these students did not demonstrate a more agile mindset, though the increased relationship between machinery and mindset may be attributed to a pedagogical approach that encourages critical thinking while learning the agile practice. In our most recent iteration of the course, we added more open-ended project constructs; for example, a 20-person team required to employ Kanban but given several degrees of freedom on how to customize the model to work for the given problem space (building a Scrum simulation) in a large team while in an academic setting. We also intend to identify activities appropriate for undergraduates to push down into the undergraduate SER316 course. Finally, we plan to investigate whether an "over-training" on agile machinery may sometimes impede developing an agile mindset, and if so how teachers overcome this barrier.

Researchers are investigating the elements of the agile mindset in professional practice [3][9][10][23]. But to our knowledge few are investigating the agile mindset in university settings, despite the growth in agile-related course offerings over the past decade. Gannod et al. (2018) [24] employs an organizational culture framework to an academic community comprised of faculty, staff, and students of an academic unit. By developing an agile mindset as part of the culture of the academic experience, students and faculty have a more productive learning trajectory. Tanaka, Saito, & Kato (2019) [25] describe a workshop-like training experience in an industry setting to instill an agile perspective in employees who are more accustomed to traditional processes like a waterfall model. Though done with a relatively small number of students in a short timeframe, the authors report positive outcomes on employee understanding of agile principles.

We intend to continue developing an agile mindset in our students by fostering teaching and learning in an agile fashion. Our work started with the Continuous Assessment Platform [26] for continuous learning feedback and now extends to more critical inquiry activities that emphasize problem-solving that eliminates waste, encourages tight feedback loops through experimentation, and asks students to reflect on agile ways of thinking and doing.

References

1. A. Przybylek, and W. Kowalski. "Utilizing online collaborative games to facilitate Agile Software Development", 2018 Federated Conference on Computer Science and Information Systems. IEEE.
2. H. van Manen, and H. van Vliet. "Organization-wide agile expansion requires an organization-wide agile mindset." Product-Focused Software Process Improvement: 15th International Conference, PROFES 2014, Helsinki, Finland, December 10-12, 2014. Proceedings 15. Springer International Publishing, 2014.
3. J. Klünder, F. Trommer, and N. Prenner. "How agile coaches create an agile mindset in development teams: Insights from an interview study." Journal of Software: Evolution and Process 34.12 (2022): e2491.
4. R. Colomo-Palaci, et al., "Competence gaps in software personnel: A multi-organizational study." Computers in Human Behavior 29.2 (2013): 456-461.
5. G. Broza, "The Agile Mindset – Making Agile Processes work", 3P Vantage Media, 2015.
6. VersionOne, CollabNet, "13th annual state of agile report." CollabNet VersionOne 13 (2019): 16.
7. R. Knaster and D.Leffingwell, "SAFe 4.5 distilled: applying the scaled agile framework for lean enterprises", Addison-Wesley Professional, 2018.
8. K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, and J. Kern, 2001. The Agile Manifesto.
9. J. Miler, and P. Gaida, "On the agile mindset of an effective team–an industrial opinion survey." 2019 federated conference on computer science and information systems (fedcsis). IEEE, 2019.
10. A. Mordi and M. Schoop, "Making it tangible-Creating a Definition of Agile mindset." ECIS. 2020.
11. V. Devedžić,. "Teaching agile software development: A case study." IEEE transactions on Education 54.2 (2010): 273-278.
12. G. Hedin, L. Bendix, and B. Magnusson. "Teaching extreme programming to large groups of students." Journal of Systems and Software 74.2 (2005): 133-146.
13. M. Kropp, and A. Meier. "Teaching agile software development at university level: Values, management, and craftsmanship." 2013 26th International Conference on Software Engineering Education and Training (CSEE&T). IEEE, 2013.
14. A. Schroeder, et al. "Teaching agile software development through lab courses." Proceedings of the 2012 IEEE Global Engineering Education Conference (EDUCON). IEEE, 2012.
15. M. Kropp, A. Meier, and R. Biddle. "Teaching agile collaboration skills in the classroom." 2016 IEEE 29th international conference on software engineering education and training (CSEET). IEEE, 2016.
16. A. Scharf, and A. Koch. "Scrum in a software engineering course: An in-depth praxis report." 2013 26th International Conference on Software Engineering Education and Training (CSEE&T). IEEE, 2013.
17. T. Reichlmayr,. "The agile approach in an undergraduate software engineering course project." 33rd Annual Frontiers in Education (FIE). IEEE 2003.
18. D.F. Rico, and H. H.Sayani, "Use of agile methods in software engineering education". 2009 Agile conference (pp. 174-179). IEEE, 2009.
19. K. Gary, T. Lindquist, S. Bansal, A. Ghazarian. "A project spine for software engineering curricular design." 26th International Conference on Software Engineering Education and Training (CSEE&T) IEEE, 2013.
20. K. Gary. "The Software Enterprise: preparing industry-ready software engineers." In *Software Engineering: Effective Teaching and Learning Approaches and Practices*, IGI Global 2009.
21. K. Eilers, C. Peters, and J. M. Leimeister. "Why the agile mindset matters." Technological Forecasting and Social Change 179 (2022): 121650.
22. S. Das and K. Gary. "Agile Transformation at Scale: A Tertiary Study". In: Gregory, P., Kruchten, P. (eds) Agile Processes in Software Engineering and Extreme Programming – Workshops. XP 2021. Lecture Notes in Business Information Processing, vol 426. Springer, Cham. https://doi.org/10.1007/978-3-030-88583-0_1.
23. N. Ozkan, and M. Şahin Gök. "Investigation of Agile Mindset Elements by Using Literature Review for a Better Understanding of Agility." Turkish National Software Engineering Symposium (UYMS). IEEE, 2020.
24. G. C. Gannod, W. F. Eberle, D. A. Talbert, R. A. Cooke, K. Hagler, K. Opp, and J. Baniya, 2018, October. Establishing an agile mindset and culture for workforce preparedness: A baseline study. In the 2018 IEEE Frontiers in Education Conference (FIE) (pp. 1-9). IEEE.
25. T. Tanaka, S. Saito, and Y. Kato. "Do Pipe Cleaners Help Software Engineers to Understand Agile Mindset?." 32nd Conference on Software Engineering Education and Training (CSEE&T). IEEE, 2020.
26. K. Gary and S. Xavier. "Agile learning through continuous assessment." In 45th Frontiers in Education Conference (FIE). IEEE 2015.