Encouraging Students to Adopt Software Engineering Methodologies: The Influence of Structured Group Labs on Beliefs and Attitudes

JEFFREY P. LANDRY
School of Computer and Information Sciences
University of South Alabama

J. HAROLD PARDUE
School of Computer and Information Sciences
University of South Alabama

MICHAEL V. DORAN
School of Computer and Information Sciences
University of South Alabama

ROY J. DAIGLE
School of Computer and Information Sciences
University of South Alabama

ABSTRACT

This study proposes that structured labs using groups can help foster individual student acceptance of software engineering methodologies. The technology acceptance model (TAM) is employed in an empirical test using students in freshman and sophomore-level programming courses. Our findings suggest that a structured group lab experience does influence a student's belief system regarding the usefulness of a software engineering methodology, leading to an individual decision to accept and use the methodology on a voluntary basis. On average, the software engineering methodology was accepted by the students sampled. We recommend that structured group labs be designed to use peer groups, reinforce successful results, and use an iterative process design with phase-by-phase deliverables.

I. INTRODUCTION

In today's environment of increasingly complex and integrated systems, software engineering methodologies are vitally important for improving the quality and predictability of software development projects. Software engineering is widely viewed as a solution to the chronic crisis of poor software quality, missed deadlines, cost overruns, and abandoned projects [1]. Success as a software engineering professional depends on a high level of competence and confidence in software engineering methodologies [2]. To achieve this competence, students must be taught effectively and encouraged to use software engineering methodologies.

The task of motivating students to accept and use software engineering methodologies is a difficult one. The aphorism that "you can bring a horse to water, but you can't make it drink" seems especially pertinent. The challenge for educators is similar to that of industry: given time pressures and other constraints, what can be done to motivate students to use scientific, systematic, and disciplined approaches to software development? Far too often, students and developers rely on *ad hoc* and undisciplined approaches [1]. This challenge can be formally stated as: how can computer science educators influence students so that they are motivated to voluntarily adopt and use software engineering methodologies?

In this paper we present a pedagogical solution to the challenge of motivating students to adopt and use a structured, software engineering methodology. Our solution was based on research in the diffusion of innovations [3], social learning theory [4], and the technology acceptance model [5], and attempted to incorporate some of the best practices derived from research on group work and interaction. Our technique of structured group labs [6] involved the use of instructor-facilitated, peer-to-peer interaction in small groups to affect student beliefs about the use of engineering methodologies.

II. BACKGROUND

A. Software Engineering Principles

Basic software engineering principles are taught early in the computer science curriculum. Most curricula present these principles as a variation of the traditional software development life cycle (SDLC). In this cycle, software development is modeled as an iterative, top-down progression of phases, each with clearly defined and verifiable deliverables. This life cycle is typically presented as a "formal" problem solving approach.

Student problem solving behavior can be classified along a continuum from impulsive to reflective [7]. Reflective problem-solving is characterized by thoughtfulness and looking back [8]. The SDLC typifies the reflective problem-solving approach. Impulsive problem-solving is characterized by a precipitous jump to an implemented solution without either sufficient reflection or thought. Although correct solutions can be conceived from an intuitive leap, standard practice prescribes reflective approaches to software development.

We believe there are at least two reasons why some students tend toward more-or-less impulsive problem-solving strategies [9]. The first reason is that students often do not have to use reflective methodologies. Even when required to submit each phased deliverable, the student has the option to behave impulsively and then

Relative advantage	The innovation is perceived as being superior to that which it supercedes.
Compatibility	The innovation is compatible with existing values, skills and work practices of potential adopters.
Complexity	The innovation is relatively difficult to understand and use.
Trialability	The innovation can be experimented with on a limited basis without undue effort and expense; it can be used incrementally and still provide a net positive benefit.
Observability	The results and benefits of the innovation's use can be easily observed and communicated to others.

Table 1. Innovation characteristics.

reverse-engineer the required submission. The second reason, and the focus of this paper, is that they care not to do so.

B. Diffusion of Innovations Perspective

In trying to understand why students might not want to adopt engineering methodologies, we looked for a theory that could explain the causal factors behind their lack of motivation. Diffusion of innovations theory has been used extensively to explain adoption behaviors [3, 10]. From this perspective, the readiness with which individuals adopt an innovation is influenced by the way potential adopters perceive the innovation in terms of its key characteristics: relative advantage, compatibility, complexity, trialability, and observability (Table 1). Students, the potential adopters, perceive that an SDLC methodology, the innovation, has varying degrees of the qualities that make it attractive for adoption. For example, a student might perceive that using a software engineering methodology provides little relative advantage over jumping to a coded solution on a simple, narrowly-focused homework problem. From a cost-benefit perspective, the relative simplicity of the assignment does not justify the overhead expense of using the prescribed methodology. A student may also believe that the SDLC represents a problem-solving approach that is incompatible, or at odds, with pre-existing values and norms established in early, simple programming assignments. Perhaps a student perceives that the formal, step-by-step methodology is difficult to understand and use. A student may be reluctant to adopt the SDLC because there is either little or no chance to experiment with it on a trial basis, and reluctance to accept the SDLC may be explained by the lack of being able to observe any results and benefits from its use.

C. Social Learning Theory

We believe that a structured group lab can be an effective means of affecting a student's perception of the key characteristics of the prescribed engineering methodology. We submit that this affect can be explained by social learning theory. According to social learning theory [4], learning is a social-cognitive process in which individual beliefs are impacted by the individual's experiences in the social context. A key idea is that of vicarious reinforcement [4] in which learners imitate the behaviors of others when these behaviors are successful and reinforced.

The structured group lab provides a peer-to-peer environment for imitating both within and between-group usage behavior. The instructor's role is to facilitate reinforcement by pointing out successful uses of the SDLC across the multiple solution variants.

D. Technology Acceptance Model

A variety of theoretical perspectives have been developed to examine the factors that influence technology usage. One stream of research has focused on intention-based models [11, 12]. The Technology Acceptance Model (TAM) is one such model [5, 13]. Based on the diffusion of innovations and social learning theories, TAM (Figure 1) explains how beliefs about a technology innovation affect individual decisions to adopt and use it.

We employed TAM to explain and measure how the structured group lab approach impacts students' motivation to adopt software engineering methodologies. Software engineering methodologies can be classified as technological innovations. Research has found that the key to getting individuals to accept a technological innovation is getting the individuals to perceive the technological innovation as *useful* and *easy to use* [5, 13]. Usefulness and ease-of-use are based on Rogers' generic characteristics (Table 1). The adapted TAM for this study is depicted in Figure 2.

One aspect of the model that is of particular interest to this study is the inclusion of external variables. "A key purpose of TAM is to provide a basis for tracing the impact of external factors on internal beliefs" [13]. External variables are interventions, such as "system

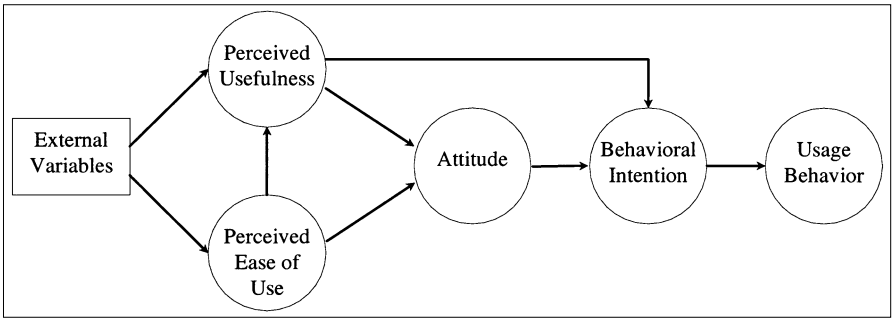

Figure 1. Technology acceptance model.

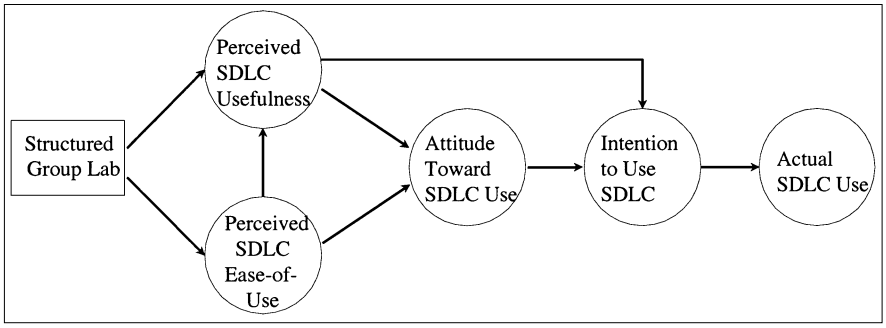

Figure 2. Adapted TAM.

design characteristics, training, documentation, and other types of support" [13], designed to influence user beliefs. A strength of this inclusion is that external variables can represent interventions made by change agents, such as either managers or educators, who are trying to influence individual acceptance of information technology. We view structured group labs as an intervention that can positively influence student perceptions about the usefulness and ease of use of software engineering methodologies, leading to increased acceptance and usage behavior. Research has shown that external variables, such as participation in training, do influence the learning process through the beliefs-attitudes-intentions pathway posited by TAM [14].

III. The Pedagogy—Structured Group Labs

The structured group lab [6, 15, 16] was designed to motivate students to adopt the prescribed reflective approach over an impulsive approach to software development through the use of peer influence and reinforcement. Peer influence and reinforcement is posited to occur both within and across groups. Within groups, students assist each other in developing the deliverable for each phase of the SDLC. Across groups, the instructor facilitates a discussion of the strengths and weaknesses of various solutions. Peer influence and reinforcement in the group lab motivates students to want-to adopt a more reflective approach in the following way. In the guided class discussions, the instructor points out to students the relative advantages of using versus not using the SDLC by using examples from the groups. While working with peers, students go through a step-by-step process of practicing the use of the SDLC so that: (1) they are comfortable with the SDLC such that they perceive the SDLC to be compatible with their own values and problem-solving norms, especially as they learn vicariously, watching the successful results of their peers get reinforced; and (2), they perceive that the SDLC is of relatively low complexity (not difficult to use). The requirement that groups submit one-at-a-time deliverables provides a trial use of the SDLC. Through facilitated inter-group discussion, the results and benefits of using the SDLC are observable by everyone in the class.

In creating an environment that addresses the motivational factors thought to influence adoption and usage, we submit that participation in a structured group lab will lead to increased individual voluntary adoption and usage of reflective problem-solving approaches. The next section discusses the methods used for assessing the impact of our particular approach.

IV. Methodology

In order to assess the influence of structured group labs on student acceptance, we used a survey design to gather data from undergraduate computer information science students enrolled in an introductory programming sequence (CS1 and CS2). The target behavior of this study was individual student use of a software engineering methodology, defined as the software development life cycle methodology, on take-home programming assignments. We were interested in the extent to which a structured group lab experience—where students practiced the SDLC in groups of three to five students—would influence usage behavior on individual, take-home, programming assignments.

A. Description of the Structured Group Lab

Early in the semester and before the structured group lab, the students were lectured on the use and importance of the SDLC methodology. The lab, conducted during a single class period, uses an iterative approach. A class of students was separated into multiple groups consisting of three to five students each. All groups worked on the same introductory programming problem, such as computing the amount of boards needed to build a fence around a building. The groups were given a specific amount of time to develop each deliverable (i.e., a design diagram or a coded solution) for each step of the SDLC. At the end of this period, the instructor facilitated a discussion of the various solutions. The instructor provided feedback to reinforce the value of that step in the SDLC in solving the overall problem. For example, if one group found an anomaly in the specifications and asked either a clarification question or made a certain assumption, the instructor pointed out the ramifications of that decision on the rest of the project, or asked if any other group had thought of that issue. This process repeated for each phase in the SDLC. The instructor's final reinforcement was the proclamation to "go out and do likewise" on the individual, take-home, programming assignment.

B. Measures

We designed and piloted a survey questionnaire to test our model (Figure 2) and the impact of the structured group lab on student motivation and adoption behavior. Items for the scales to measure each of the variables listed in Table 2 were derived from prior studies of TAM [13, 17]. From TAM, usefulness and ease-of-use are retained as the salient beliefs proposed to influence an individual's attitude about adoption. These constructs were operationalized as perceptions of usefulness and ease-of-use as formed in the structured group lab

Variable	Definition
perceived usefulness in lab	beliefs that using SDLC actually increased performance in a lab assignment
perceive ease of use in lab	beliefs that using SDLC was relatively free of effort in a lab assignment
attitude towards SDLC use	the overall favorableness or unfavorableness of an individual's feelings toward using SDLC on a given task

Table 2. Constructs used in this research.

	# Items	Mean	Standard Deviation	Reliability
Usage Behavior	4	2.65	0.37	0.849
Behavioral Intent	3	2.30	0.80	0.977
Attitude	4	2.37	0.40	0.921
Usefulness	3	2.90	0.15	0.897
Ease of Use	4	2.39	0.85	0.961

Table 3. Scale characteristics.

environment. Since this study was interested only in the impact of the structured group labs on student beliefs about using SDLC, and not on the impact of either readings or the instructor's lectures, the belief constructs were operationalized so as to isolate student perceptions to their experiences in the group lab. All items were formulated using a seven-point semantic differential scale with bipolar adjectives. Behavior items represented self-reported usage.

C. Pilot Study

A pilot study was conducted to assess the validity and reliability of all measurement instruments and to make a preliminary assessment of the model relationships. Twenty-five CIS students enrolled in an introductory programming (CS1) course were administered the instrument. Item analysis was used to streamline the scales to either three or four items per construct. Items with low reliability were either dropped or revised.

D. Data Collection

The final instrument was administered to 34 students. The data were collected in two waves in order to reduce the effects of common method variance. Shortly after participating in the structured group lab and before working on an individual assignment, the students responded to survey questions on all variables except the dependent variable. The variables measured in the first phase included student beliefs, determinants of intention and their intention to use the SDLC on programming assignment. After completing the individual, take-home, programming assignment, students were administered the dependent variable items to assess the extent to which they used the SDLC.

E. Construct Validity

The revised scales exhibited a high level of construct validity. The scales were subjected to iterative factor analysis to assess construct validity. The analysis of the factor loadings λ for each construct revealed significant loadings as predicted. The factor loadings ranged from 0.745 to 0.996. Internal consistency, or reliability, was assessed using Cronbach's Alpha coefficient. Overall, the scales exhibited high levels of reliability. The reliabilities ranged from 0.849 to 0.977. A summary of the scale characteristics is provided in Table 3.

V. Results

Scale limits for all of the measurement constructs ranged from 1 to 7, with "1" at the high end of the scale, "4" at the midpoint, and "7" at the low end. The mean score for usage behavior of 2.65 indicates that students, on average, did consistently apply the SDLC on their take-home assignment. Mean scores for student beliefs about the ease of use and usefulness of the SDLC, attitude toward using the SDLC, and intentions to use the SDLC were also high, ranging from 2.30 to 2.90 (Table 3).

The overall fit, predictive power, and path significance for the model (Figure 3) were examined using Amos 3.6, a structural equation modeling software tool [18]. Overall, the fit statistics suggest that the model fits the observed data quite well ($\chi^2 = 3.74$, $p = 0.442$). The χ^2 is not significant at any alpha level, and all the other fit statistics are well within range. The model accounts for 33% of the variance in behavior, 60% of the variance in intention, and 84% of the variation in attitude.

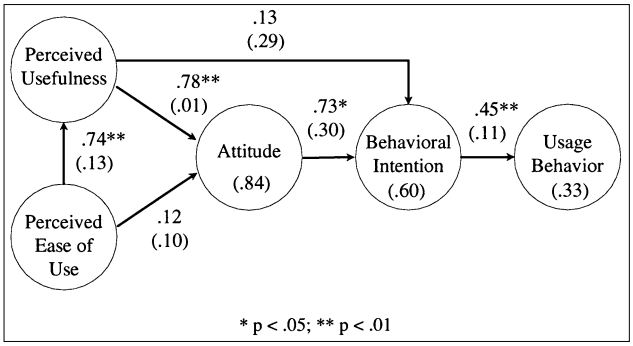

Figure 3. Model with path coefficients.

The hypothesized relationship between perceived usefulness and usage was significant along the beliefs-attitudes-intentions-behavior path. These results are consistent with prior TAM studies, which suggest that perceived usefulness indirectly influences usage and intention through attitudes. The level of statistical significance is somewhat impressive in light of the small sample size (N = 34). The direct effect of perceived usefulness on intention was not significant. This is not surprising given that this path was included to account for intentions formed in an organizational setting above and beyond the positive or negative feelings evoked toward the behavior. Students in a classroom are not exposed to the same reward structures found in most organizational settings. The direct effect of perceived ease of use on attitude was not significant. Prior studies of TAM have found that the influence of perceived ease of use is equivocal and often mediated by perceived usefulness. Our findings confirm this observation. The significant path from perceived ease of use to perceived usefulness suggests the influence of the perceived ease of use on attitude is mediated by perceived usefulness.

VI. Discussion

In this study we examined the influence of structured group labs on student acceptance and usage of software engineering methodologies by measuring student usage behavior of a software engineering SDLC on take-home programming assignments. Our findings suggest that the structured group labs motivated students to adopt voluntarily and use software engineering methodologies on their individual programming assignments by influencing their beliefs about the usefulness and ease-of-use of the SDLC.

The emphasis of the structured group lab was not so much how well individuals and groups performed in the lab, but how the lab experience affected their motivation to use the methodology in completing an individual programming assignment. The implication of this study is that a structured group lab can influence student beliefs about a prescribed software engineering methodology such that the student, as an individual, will want to use the methodology. The results indicate, furthermore, that the motivational impact of structured group labs can result in a high level of acceptance of a software engineering methodology. These results are important because the promise of software engineering methodologies can only be realized if students believe that these approaches work and use them accordingly.

We believe that structured group labs have the potential to work for a variety of software engineering methodologies. The following key recommendations for designing a structured group lab are offered: use peer groups for vicarious learning; reinforce successful results for the observability of specific benefits of using the methodology; and structure the lab iteratively, with phase-by-phase deliverables, so that students can start to become more comfortable with and proficient with the methodology through trial uses. If these guidelines are followed, our results suggest that student adoption will be affected through the beliefs-attitudes-intentions causal chain, resulting in the desired motivational effect.

Furthermore, our study provides educators with a means of evaluating the effectiveness of their structured group labs. Although the TAM-based survey questions used are specific to the SDLC used in this study, the items can easily be modified to fit any prescribed engineering methodology. Educators can use the modified questionnaire to track changes in student beliefs, attitudes, intentions and behaviors over a semester, thereby assessing the impact of the structured group labs on student motivation.

References

[1] Gibbs, W.W., "Software's Chronic Crisis," *Scientific American*, September 1994, pp. 86–95.

[2] ACM/IEEE-CS Joint Curriculum Task Force. *Computing Curricula* 1991, February 1991.

[3] Rogers, E.M., *Diffusion of Innovations*, The Free Press, New York, 1995.

[4] Bandura. A., *Social Learning Theory*, Prentice Hall, Englewood Cliffs, New Jersey, 1977.

[5] Davis, F.D., "Perceived Usefulness, Perceived Ease of Use, and End User Acceptance of Information Technology," *MIS Quarterly*, vol. 13, 1989, pp. 318–339.

[6] Pardue, J.H., M.V. Doran, and H.E. Longenecker, Jr., "Student Perception of Benefits of a Structured CS1 and CS2 Lab Environment," *Journal of Computer Information Systems*, vol. 34, no. 4, 1994, pp. 40–43.

[7] Messer, S., "Reflection-Impulsivity: A Review," *Psychological Bulletin*, vol. 83, no. 6, 1976, pp. 1026–1052.

[8] Polya, G., *How to Solve It*, Princeton University Press, Princeton, NJ, 1957.

[9] Merrienboer, J., "Relationship Between Cognitive Learning Style and Achievement in an Introductory Computer Programming Course," *Journal of Research on Computing Education*, Winter 1988, pp. 181–186.

[10] Leonard-Barton, D., "Implementing Structured Software Methodologies: A Case of Innovation in Process Technology," *Interfaces*, vol. 17, no. 3, 1987, pp. 6–17.

[11] Ajzen, I., "The Theory of Planned Behavior," *Organizational Behavior and Human Decision Processes*, vol. 50, 1991, pp. 179–211.

[12] Ajzen, I., and M. Fishbein, *Understanding Attitudes and Predicting Social Behavior*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1980.

[13] Davis, F.D., R.P. Bagozzi, and P.R. Warshaw, "User Acceptance of Computer Technology: A Comparison of Two Theoretical Models," *Management Science*, vol. 35, no. 8, 1989, pp. 982–1003.

[14] Agarwal, R., "Are Individual Differences Germane to the Acceptance of New Information Technologies?" *Decision Sciences*, vol. 30, no. 2, 1999, pp. 361–391.

[15] Daigle, R.J., M.V. Doran, and J.H. Pardue, "Group Zig Zag: An Extension of Myers's Zig Zag Model," *Journal of Psychological Type*, vol. 48, 1999, pp. 34–41.

[16] Daigle, R.J., M.V. Doran, and J.H. Pardue, "Integrating Collaborative Problem Solving Throughout the Curriculum," *Proceedings of the Twenty-Seventh SIGCSE Technical Symposium on Computer Science Education*, February 1996, Philadelphia, PA.

[17] Taylor, S., and P.A. Todd, "Understanding Information Technology Usage: A Test of Competing Models," *Information Systems Research*, vol. 6, no. 2, 1995, pp. 144–176.

[18] Arbuckle, J.L., *Amos*, Smallwaters Corporation, Chicago, IL, 1997.

Author Biographies

Jeffrey P. Landry is Assistant Professor of Computer and Information Sciences (CIS) in the School of CIS at the University of

South Alabama. His research interests include managerial and behavioral aspects of software development, software engineering, information technology innovation diffusion and infusion, and CIS education and curriculum development. He has published in numerous conference proceedings and in the *Journal of Information Systems Education*. His teaching interests include visual/event-driven programming, database programming, human-computer interaction, software project management, and research methods. He received his Ph.D. degree in Information and Management Sciences from the Florida State University in May 1999. He previously worked in the commercial software development sector for eight years as a software engineer, project manager, and software department manager.

Address: University of South Alabama, School of CIS, FCW-20, Mobile, AL, 36688; telephone: 334-461-1596; fax: 334-460-7274; e-mail: landry@cis.usouthal.edu.

Dr. Harold Pardue is Associate Professor of Computer and Information Sciences in the School of Computer and Information Sciences at the University of South Alabama. He received his Ph.D. in MIS from the Florida State University in 1996. His current research interests include trust in e-commerce environments, web-based HCI, open source software development, and CIS education and curriculum development. His articles have been published in the *Journal of Computer Information Systems,* the *Journal of Psychological Type, System Dynamics Review,* and *The Engineering Economist.*

Address: University of South Alabama, School of CIS, FCW-20, Mobile, Alabama, 36688; telephone: 334-461-1600; fax: 334-460-7274; e-mail: jpardue@jaguar1.usouthal.edu.

Dr. Michael V. Doran is Professor of Computer and Information Sciences in the School of Computer and Information Sciences at the University of South Alabama. He is also the Coordinator of the Computer Science specialization. He received his Ph.D. in Computer Science from Tulane University in 1989. His current research interests include CIS education and curriculum development, software engineering, robotics, and real-time systems. He has received funding from NSF for curriculum development and robotic equipment. He has published in numerous conference proceedings, having been awarded Best Paper at ISECON on three occasions. He has also published in the *Journal of Computer and Information Systems,* the *Journal of Psychological Type* and the *Journal of Information Systems Education.*

Address: University of South Alabama, School of CIS, FCW-20, Mobile, AL, 36688; telephone: 334-460-6390; fax: 334-460-7274; e-mail: doran@cis.usouthal.edu.

R.J. Daigle is Professor of Computer and Information Sciences (CIS) in the School of CIS at the University of South Alabama. He is also the Coordinator of Information Systems and the Director of CIS Graduate studies. His research interests include curriculum development and database theory. He has published in numerous conference proceedings, the *Journal of Information Systems Education*, and the *Journal of Psychological Type*. He received the Ph.D. in Mathematics from the University of Georgia in 1972.

Address: University of South Alabama, School of CIS, FCW-20, Mobile, AL, 36688; telephone: 334-460-6390; fax: 334-460-7274; e-mail: daigle@cis.usouthal.edu.