Assessing Active Alternatives for Teaching Programming

Sandra Poindexter
Northern Michigan University, Marquette, MI, USA

spoindex@nmu.edu

Executive Summary

Active learning and cooperative learning are two alternatives to the traditional lecture/lab approach to teaching software development. Evidence supporting use of these learning strategies in Computer Information Systems (CIS) began to emerge in the mid-1990s as faculty sought ways to improve student understanding of programming concepts, reduce the level of frustration, and better prepare students for jobs based on teamwork.

Shifts in the competencies of incoming students have influenced the type of material covered in the standard class period. Though a higher percentage of students may enter knowing the basic mechanics of the computer, it remains difficult for them to effectively apply that skill beyond their personal interests of music, games, and surfing. Used to an advantage, these changes in student competencies can provide more time for "why" topics in place of "how to" topics. Technology integration enables the shift to occur more efficiently, but alternatives to a pure lecture mode of course delivery may be worthwhile.

A study with active and peer learning strategies that was conducted at this mid-west USA institution involving software development is reported here. The results of this study indicate that some lecture is necessary, structure is critical, and careful attention to group processes and facilitation is needed. The rewards are that active exercises shorten the learning cycle and improve problem-solving abilities, attitude is significantly improved, and interpersonal skills are gained. Using literature and empirical research this paper assesses the value of active and peer learning environments in programming courses.

Keywords: innovative teaching, active learning, cooperative learning, peer learning, problem-based learning, programming, software development, teaching and learning, web-enhanced

Literature Review

History and Definitions

In the introductory chapter of their book on cooperative learning, Millis and Cottel present an excellent history and rationale for learning as a social affair. In the early 1990s researchers, such as Slavin, Astin, and the team of Johnson, Johnson and Smith, synthesized studies conducted during the 1980s (Millis & Cottell, 1998). Their conclusions continue to ring true as cooperative learning has been soundly researched across disciplines; learning within the context of a group of peers is at least as effective as lecture for content knowledge gain, attitudinal changes are positive, and students gain important interpersonal skills. The terms "collaborative" and "cooperative" are often used interchangeably. When a difference is noted, "cooperative" is usually considered a subset, or stepping stone, to the broader "collaborative." For this paper, cooperative learning consists of a structured environment wherein the instructor is a constant facilitator

amidst student peer groups whose members work on group processing of specified problems, are responsible for preparation as individuals, and employ team skills. The term "peer learning" will be used to mean the same. This definition does not insist upon group grading or prescribed group composition.

Active learning literature emerged as a specific subject heading in the early 1990s, often cross-referencing earlier cooperative learning studies. However, the two are not the same. Though active learning is often a component of cooperative learning when peer groups work together during class, active learning can be conducted without groups. The most often quoted definition of active learning, coined by Bonwell and Eison in 1991, is "anything that involves students in doing things and thinking about things they are doing" (Bonwell & Eison, 1991). It can include short and frequent pauses in lecture for reflection, sharing, summarizing, and extending, and works well in large lecture halls or small classrooms. Entire class periods can be devoted to an active approach with students working individually or in groups. Structure of the environment (class meetings, the curriculum, and assessment) is a factor deemed critical to the success of active learning (Miller, Groccia, & Wilkes, 1996).

Evidence of Change in Computing Disciplines

In the disciplines of CIS and CS software development courses have traditionally been taught as lecture or lecture + lab/discussion, using theory-based textbooks with examples, and assessed with individually assigned programming problems and tests. In the mid 90s self-paced, highly annotated tutorials became textbooks options. A variety of factors impact the learning environment of computer information systems (CIS) and computer science (CS) courses: subject complexity, student culture, faculty composition and philosophy, technology integration, institutional infrastructure, a dynamic subject, and high industry expectations. Given this setting, educators may be reluctant to risk change an instructional process in place for years.

While student teams were sometimes used for out of class work on sizable projects, cooperative and active learning, as defined in this paper, were generally not introduced until the mid 1990s. An early presentation on teaching approaches in 1994 by Dutt described use of programming teams, a combination of traditional lecture and group exercises, and group testing in a beginning programming course. Students assessed it as a positive experience and realized team synergy (Dutt, 1994). In 1996 McConnell speaks of active learning in CS using modified lectures (brief lectures interspersed with reflective questions or sharing), algorithm tracing exercises, and physical activities to simulate computer processing. Results of his controlled, multivariate experiments in an introductory programming course were that active learning students had higher exams scores. Students exposed to both active learning and group activities seemed to be further helped by a marginal amount (McConnell, 1996). In the same year, Cordes and Parrish described their software development classroom redesigned from a traditional lecture to a more active learning environment. Using mini-lectures, active exercises, and peer learning they report that the shift in student attitude was remarkably positive and performance on the in-class exercises was outstanding (Cordes & Parrish, 1996). Later in the decade, Granger and Lippert documented the steps for, and effects of, peer learning in an introductory programming course. They modified lecture/lab to incorporate a weekly peer learning session where students worked on exercises in groups of three. Earlier student perceptions that programming was "dry, boring, and tedious" were replaced with enjoyment, engagement, and participation in the class and subject. They found the exercises improved problem-solving abilities and knowledge gain, and the teamwork fostered growth in communication and collaboration skills (Lippert & Granger, 1997).

Granger and Lippert (1999) later expanded their use of peer learning across the IS curriculum and are supported by other researchers in similar IS or related technical studies. Hyper-link teaching was documented in 1997 by Purao to foster active learning in a systems analysis course through modification of a typical lecture slide presentation with non-linear anchor points for discussion. This approach retained the positive factors of presentation slides and countered the passivity associated with sitting and watching

slides (Purao, 1997). Buffington specifically measured the effect of self-directed teams in a computer literacy course, reporting in 1998 that students supported the value of working in teams (Buffington, 1998). To actively engage students in an MIS course Mukherjee used active, but not peer, learning and found improved student attendance, less boredom, more critical questions, more motivation, and better understanding. In addition, Mukherjee reports in 2000, students perceived the class sessions were more productive than in a control group that did not use the active learning strategy. In 2001, Neufeld and Haggerty concluded from an empirical study on collaborative team learning in a database management course that team learning fostered high team performance that exceeded individual performance and built team skills while concurrently conveyed technical knowledge.

A Quantitative Study: Introductory Programming

Study Description

The review of literature provides a history of variations in teaching programming that were singular (one variation), anecdotal, or did not heavily use technology in the traditional classroom. To assess the combined effectiveness of active and peer learning approaches in an introductory programming course, a yearlong experiment was conducted in a controlled setting supported by a laptop environment. A related question facing faculty is whether a threshold exists in the levels of learning strategy alternatives needed to produce a satisfactory return on their preparation time investment. Is there a point at which a higher investment of time and effort no longer produces significant additional gains in student learning or attitude towards learning?

Varying the proportions of lecture, active, and peer learning over four offerings of a two-credit, eight-week Visual Basic course enabled a more distilled look at impact. Consecutive offerings were logistically necessary in order to control for instructor. Taken by CIS majors during their late freshman or early sophomore year, this course serves as a gateway to the upper division courses and has a diverse group of students. The enrollments during the study were 23, 23, 24, and 24 students, respectively. In an attempt to control variables between student groups and times offered, and to collect any potential correlating data, pre-test and post-test knowledge data, demographics, pre-attitudinal information, learning styles, and personality types were gathered from each group. The syllabi, scheduling book listing, and rigor of the class were consistent across all sections. Methods of assessment were programming assignments, written content exams, programming skills test, and participation. For all sections, a course website posted a syllabus, outline, handouts, files, and assignment instructions. This paper presents a summary of the study. Complete details, statistical analysis, and limitations are available in an earlier report (Poindexter & Allen, 2001).

The first phase reflected a traditional instructional delivery held in a room with rows of desks facing forward. Though each desk had a computer, it was used for downloading presentation slides or handouts rather than for programming instruction. Using large screen projections of prepared slides, class time was 100% lecture unless a student asked a question or offered comments. The textbook tutorials and assignments were done outside of class with no student collaboration permitted – all work was done individually.

In Phase II, a modified lecture approach of 50% lecture and 50% ungraded, active practice exercises was used. Peer interaction in and out of class was encouraged, but it was not required nor were the benefits of teamwork explained. The room was identical to that of Phase I, rows facing forward with each student having a computer. During a typical class, there would be an introduction of the day's topics followed by a short lecture using presentation slides and sample Visual Basic programs to cover the most complex portions of the textbook material or to give an overview to put the textbook items into a broader perspective. Either during the presentation or after it, students would be told to download an exercise to be completed immediately. Typically, the exercise involved modifying or completing a pro-

gram shell according to general guidelines; specific steps to complete the work were not provided, reinforcing to students the need for preparation prior to class. Students chose to work alone or with a person sitting next to them – there were no assigned pairs. Completed solutions were emailed as file attachments to the instructor as a record of attendance and participation, but not graded. Preparation time increased for two reasons. First, it was necessary to choreograph the mix of lecture, pause, and exercise to fit the class time slot. Annotated presentation slides were added to the course website as outside lecture in order to free class time for the active learning exercises. Second, lecture demonstration files had to be converted for use as in-class exercises requiring careful naming, uploading to the server, and linking for student access during class. General instructions had to be accurate to prevent student confusion. Instructional energy required during class time increased to handle problems as students worked alone. The emphasis in Phase II was active, rather than peer, learning.

Phase III was held in a group conference-room style classroom with tables seating four, facing each other, and full network and electrical connections for a laptop computer at each seat. Class time consisted of 40% lecture and 60% active exercises. Benefits of peer learning were explained to the class and cooperative learning was expected during class time. The Phase II lecture and active learning mix was supplemented with peer learning. Students worked with others at their table and participation included evidence of helping peers in and out of class time, such as emails between students asking for help or clarification. Collaboration was encouraged, but not required on graded assignments done outside of class and "good" solutions were posted to the website for peer review. Some preparation time beyond Phase II was required to refine lecture slides and in-class exercises to shift content coverage from the former to the latter. Class time instructional energy decreased from Phase II as peer learning took hold and students helped each other. The emphasis in Phase III was to blend active and peer learning.

The final Phase IV expanded interactivity and creativity with 30% lecture and 70% active, peer learning. A casual learning environment was created where people felt free to talk and walk around to view the progress of other groups. Assignment creativity and solution variation were encouraged. Students could alter assignments or expand exercises by programming additional features or improving the look of the program's screens. Being creative earned more participation points, but did not add to the assignment score. Students were expected to work on assignments in pairs to continue the peer learning outside of class.

Study Outcomes

The letter grades, average overall grade percentage, and component scores are means of determining student performance. In all four phases, students were individually given three tests. Two were written exams on terminology, programming concepts, and syntax recognition and error resolution that did not involve use of the computer. In the third test during the last week of class, students applied their knowledge by programming a working solution for a new program specification. Table 1 provides the numerical results of letter grade distribution.

Grade	I	II	III	IV
A	1	7	7	6
B	8	6	6	6
C	4	1	4	4
D	2	1	5	5
F	4	3	1	1
Withdrew	4	5	1	2
Total # students	23	23	24	24

Table 1. Distribution of Overall Grades by Phase

In the Phase I traditional lecture, 39% of the students received an A or B while in all the other phases using non-traditional learning the range was 50-56%. The data indicates that letter grades were higher as more interactivity was added. The overall grade was broken down into its three components – assignments (40%), participation (10%), and tests (50%) – and is graphed in Figure 1. Phase I has lower scores across all three components with the other three phases similar to each other. Figure 2 indicates that there was not a wide variation in learning terms and basic concepts (written tests) between the phases - lecture worked equally as well as the active variations, confirming the findings in the literature review. However, students in all phases with active learning and peer learning scored significantly higher on the software skills test that involved active problem solving. It seems the extra practice and on-site assistance made them better prepared for applying the content knowledge.

Retention data was kept initially as part of university record, however, the data appeared to form a pattern related to the study. Fewer people withdrew as peer learning was added to the course. In Phases III there was an 8% withdrawal rate and Phase IV had a 12% withdrawal versus 35% in Phases I and II. The value of "W" measured the number of students who withdrew from the course after completing at least ½ of the course's duration. When these counts are added to the counts of failures, they become more significant as active learning was increased.

Figure 1. Phase Comparison by Average Grades

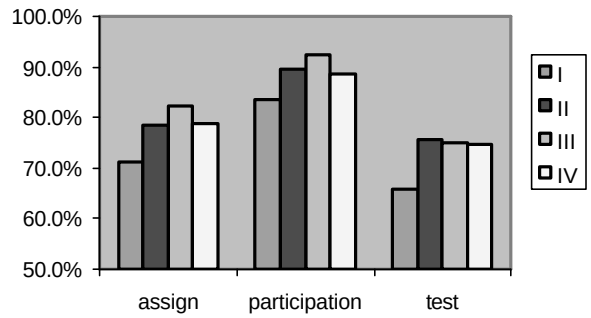
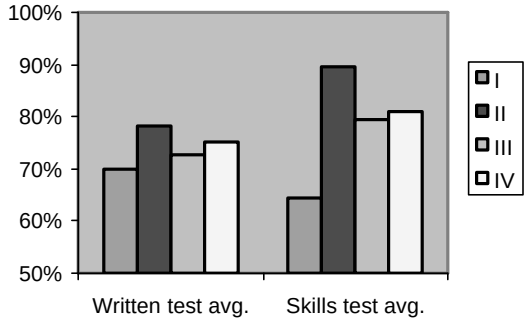


Figure 2. Phase Comparison by Test Scores



Student perceptions were measured via an 11-question course evaluation instrument including satisfaction with the course and satisfaction with their own contribution. While not statistically significantly different, the same pattern repeats from the grade outcomes – the traditional approach ranked lower in satisfaction than the other three phases for both the course and participation. These numbers correlate with the written comments. In Phase I, numerous comments were given such as "it was not very exciting and did not hold my attention," "needs better examples and showing how to do specific things," "if there was more time, maybe get into groups and do a simulation of a real business programming project," and "more student interaction needed." Most students liked learning to program, but not the method of delivery. Comments from students suggested they were less motivated in lecture. Apathy, clock-watching, boredom, and fatigue existed. Phase II was essentially void of written comment, however, class time was more alive. Some students appeared uncomfortable and physically distanced themselves from others to avoid having to pair up with another student. Phases III and IV, had numerous positive comments such as "The class was fun and I learned a lot," "I learned a lot in a short period of time," and "learned great organization skills in respect to the course."

Discussion

The primary hypothesis being tested was that incorporation of active and peer learning has no effect on grade. With the small sample sizes, the study lacked the statistical power to determine whether or not

variations in teaching methods between phases had an effect on overall grade percentage. However, when looking at the letter grades and the problem-solving skills test, the case is more strongly in favor of the active and peer learning strategies. This confirms the literature review that alternative active and peer learning strategies are at least as good as lecture for content knowledge and is better for applying knowledge to a new situation.

The distinctions between this study's setting and the tutorial-based lab setting used for many computer courses are the peer learning emphasis rather than independent lab work, and active computer exercises integrated within every lecture rather than having a designated lab day. Programming requires practice and students are not likely to spend time outside of class doing "extra" ungraded exercises. They are, however, more apt to read and outline the text content and complete graded assignments. This model played upon those tendencies by reversing the activities in and out of class. Students quickly realized that they could not do the in-class exercises if they had not read the text – there would be no lecture to make up for not doing so and they became accountable. This accountability confirms the literature findings on the benefits of active learning. The elevated course evaluations confirm cooperative research findings: it is attitude towards learning that is most positively impacted by peer and active learning, and knowledge gain is at least as good.

The issue of retention was carefully analyzed. Is it better to retain marginal students or let them fail early, forcing them to retake the course? The answer partly depends on the institutional GPA calculations for repeated courses, demand levels for the academic program and classes, whether the student changes majors, or intends to stay at the institution. In the active, peer environment, students may have felt more connected to the class, less singularly frustrated, and more academically confident. If a student has enough motivation to persevere, s/he may reach a higher level of learning in the long run. When future tasks prove difficult, a benchmark exists by which to measure the probability of success. If concern for subsequent classes exists, a curriculum can force a repetition of a class if a student passes with a poor grade. Encouraging struggling students to stay in a class will at least provide more contact with the subject matter in the event of a class repetition.

The group table setting for the last two phases significantly increased discussion and peer learning. Physical movement increased – students got up to see how others were doing or to help someone else. Noise level increased – students were more willing to ask one another a question, asking the instructor after none of the peers knew the answer. Clock watching, particularly in the peer learning settings, decreased – students were not preoccupied with how much time was left. It is probable that by physically facing their peers rather than the front of the room and verbal reminders to assist each other, students were able to alter their solitary behavior. Since there was no difference between the phases in student population for learning styles, personality types, or attitudes, it leads one to believe the learning environment affected the outcomes.

The second hypothesis regarding a threshold of effort that a faculty member must make in the adoption of active and cooperative learning for programming courses for success to occur seems to hold true on the surface. There is a preparation cost to shift materials from lecture to active learning exercises to at least the 50% level (Phase II). Once made, there may not be much to gain from additional efforts. At a deeper level, shifting into peer learning (Phase III) required an effort in relinquishing control in order to gain attitudinal improvement evidenced in written comment rather than scores. Phase IV did not seem to raise either quantitative scores or attitudes significantly. It is left to instructors to find a measure for themselves by retaining records of their effort and student outcomes. When a positive outcome becomes evident, an instructor may find it worthwhile to pace at that level. One factor to include in this formula is the perception of true success for both faculty and students. It is not easy to quantify, but easy to see and feel.

In sum, this study confirms and enhances the literature reviewed for active and peer learning in technical courses by McConnell, Cordes and Parrish, Granger and Lippert, and Mukherjee. Some lecture is still needed, structure is critical, active exercises shorten the learning cycle and improve problem-solving abilities, careful attention to group processes and facilitation is needed, attitude is significantly improved, and interpersonal skills are gained.

Technology Supports Active Alternatives

Restricted to a once-a-week lab day, faculty are less apt to ask students to listen and try than if computers are available for each student or student group at all times. The study documented here was conducted at a mid-west, regional institution of approximately 8,000 students under a campus-wide laptop policy. The mandatory program standardized a set of technology tools (laptop, software, and Internet access) for all full-time students and faculty, provided all classrooms with a network port for an instructor workstation and 30 classrooms with electrical and network outlets at each student seat. Using their own laptop gave students more confidence in computer operation. It opened the door for a significant increase in active computer exercises during any and all class times, and ensured that all students had total access to computing hardware and software to download slides, handouts, and complete assignments without going to a lab. (Northern Michigan University, n.d.).

The new student generation enters college with a higher degree of computer literacy, albeit used for music, games, communicating, and surfing the Web. Faculty can capitalize on this general knowledge to move quickly to a higher level of programming by asking students to work out logic problems on the computer, purposely entering a logic or syntax bug to experience the results. It is not the same for the instructor to convey an event, or even for students to predict one, as it is for them to immediately test the outcome on their computer. Technology integration enables the shift from facts to application to occur more efficiently and effectively.

While individual laptops at every class section was an unquestionable asset to this study, active exercises can be tailored to share resources that do exist, such as grouping students with one laptop available per group. It is likely that computing students will opt to purchase them, or curriculum or grant funds may be tapped to equip “active learning” classrooms. Increasingly, laptop use and laptop policies have the potential to significantly alter the pedagogy in the programming classroom. The 2001 Campus Computing Survey reports up to 20% of university and college students own a laptop computer (Green, 2001). A tracking of institutional laptop policies by Brown, Director of Associated Colleges of Central Kansas, lists 186 colleges and universities with laptop initiatives as of May 2003 (Brown, 2003). Laptop technology complements the cooperative and active learning environments so effectively, enhances the technical knowledge of computing students, and takes advantage of campus technology initiatives that it is a trend well suited to the computing discipline.

Risk Versus Rewards

Faculty may welcome the chance to improve the learning environment in their classrooms if they could predict the return on their investment of their limited resources. It does not appear necessary to go to elaborate lengths in setting the classroom environment – gains and attitudinal shifts can be derived from modest introductions of active and peer learning strategies. The more students are engaged, the higher the instructor self-fulfillment. Depending upon the personality of the instructor, fulfillment may be benefit enough to cover the additional costs of preparing group activities and the change to a mentor/ consultant role. Documented pitfalls are too little structure, lack of communication to students on the purpose and procedures of the new learning environment, and student-student interpersonal problems that go unresolved. A common error of faculty new to active learning is to forgo structure which ends in chaos. For most faculty, it takes several semesters of trial to successfully shift paradigms and an initial

drop in student evaluations might occur. Research and this study show there is not a lot to lose, and definitely some to gain by incorporating alternative teaching strategies.

There is also reward on the side of industry who highly values the non-technical skills of new graduates such as the ability to learn and teamwork. In a study comparing employer expected (desired) and actual valuation of 19 technical and 18 non-technical skills of new IS graduates, Cappel found that the non-technical skills had the most room for improvement. Problem-solving, oral communication, motivation, initiative, and attention to detail were at the top of the deficiency list (Cappel, 2001). A related study by Tang, Lee, and Koh measured the gap between educator's perceived required skills and actual skills attained by graduates. As with the employer assessment, more deficiencies were found in non-technical skills than technical knowledge. Interpersonal communication and behaviors, critical and creative thinking, and motivation and independent learning all had significant gaps and all were in the top seven skills desired by employer (Tang, Lee, & Koh, 2001). These non-technical traits are best addressed by active classroom alternatives without the risk of foregoing the required technical skills.

Resources

Very detailed and practical suggestions are outlined by Buckenmyer for using teams with class activities to overcome typical problems associated with teams (Buckenmyer, 2000). More guidelines and a student handout on self-managed teams are provided by Van Slyke, Trimmer, and Kittner (1997). Edited books by Foyle (1995) and Sutherland and Bonwell (1996) on active approaches for higher education combine the theory and application. Felder from science and Brent from education have collaborated on numerous practical articles on the road to student-centered environments, their procedures, pitfalls, and payoffs (Felder &Brent, 1994, 1996). All these resources stress the importance of convincing students that the team approach has value to them – buy in is critical to the success. Three websites provide assistance to faculty in technical programs: McConnell's Active Learning < http://www-cs.canisius.edu/~mcconnel/active_learning.html >, Felder's resources < http://www.ncsu.edu/felder-public >, and National Institute for Science Education College Level One for collaborative learning < http://www.wcer.wisc.edu/nise/cl1/ >.

References

Bonwell, C., & Eison, J. (1991). Active learning: Creating excitement in the classroom. *ASHE-ERIC Higher Education Report No. 1*, Washington, DC: The George Washington University, School of Education and Human Development.

Brown, R. (2003, May). *Universities and colleges with laptop and notebook initiatives*. Retrieved May 2003, from http://www.acck.edu/~arayb/NoteBookList.html

Buffington, J. (1998, December). Self-directed teams in the introductory information systems course: Lessons learned. *Proceedings to the 13th International Academy for Information Management Annual Conference*. Helsinki, Finland. (ERIC Document Reproduction Service No. ED 431420)

Buckenmyer, J. (2000, Nov-Dec). Using teams for class activities: Making course/classroom teams work. *Journal of Education for Business, 76* (2), 98-107.

Cappel, J. (2001-2002, Winter) Entry-level IS job skills: A survey of employers. *Journal of Computer Information Systems, 42* (2), 76-82.

Cordes, D., & Parrish, A. (1996, June). Active learning in technical courses. *Proceedings of the 17th Annual National Educational Computing Conference*. Minneapolis, MI. (ERIC Document Reproduction Service No. ED 398884)

Dutt, J. (1994, December). A cooperative learning approach to teaching an introductory programming course. *Proceedings of the ninth International Academy for Information Management, Las Vegas, NV*, 225-232.

Felder, R., & Brent, R. (1994). Cooperative learning in technical courses: Procedures, pitfalls, and payoffs. *National Science Foundation*. (ERIC Document Reproduction Service No. ED 377038)

Felder, R., & Brent, R. (1996). Navigating the bumpy road to student-centered instruction. *College Teaching, 44*, 43-47.

Foyle, H. (Ed.) (1995). *Interactive learning in the higher education classroom: Cooperative, collaborative, and active learning strategies*. Washington, DC: National Education Association.

Granger, M., & Lippert, S. (1999) Peer learning across the undergraduate information systems curriculum. *The Journal of Computers in Mathematics and Science Teaching, 18* (3), 267-285.

Green, K. (2001, October). The 2001 campus computing survey. *Proceedings of the EDUCAUSE 2001 Annual Conference*. Indianapolis, IN.

Lippert, S., & Granger, M. (1997, December). Peer learning in an introductory programming course. *Proceedings of the 12th International Academy for Information Management Annual Conference*. Atlanta, GA. (ERIC Document Reproduction Service No. 422924)

McConnell, J. (1996). Active learning and it use in computer science. *Association for Computing Machinery SIGCSE Bulletin*, 28:SP1, 52-54.

Millis, B., & Cottell, Jr., P. (1998). *Cooperative learning for higher education faculty*. Phoenix, AZ: American Council on Education Oryx Press.

Miller, J., Groccia, J., & Wilkes, J. (1996). Providing structure: The critical element. In T. Sutherland and C. Bonwell (Eds.), *Using Active Learning in College Classes: A Range of Options for Faculty* (pp. 17-30). San Francisco, CA: Jossey-Bass.

Mukherjee, A. (2000, Spring). Effective user of in-class mini case analysis for discovery learning in an undergraduate MIS course. *Journal of Computer Information Systems, 40* (3), 15-23.

Neufeld, D., & Haggerty, N. (2001, Fall). Collaborative team learning in information systems: A pedagogy for developing team skills and high performance. *Journal of Computer Information Systems, 42* (1), 37-43.

Northern Michigan University. (n.d.). TLC Initiative. Retrieved July 15, 2003, from http://www.nmu.edu/laptops.htm

Poindexter, S., & Allen, D. (2001, Fall). Customizing the classroom learning environment – A phased experiment. *Issues in Information Systems, 2*, 364-370

Purao, S. (1997, December). Hyper-link teaching to foster active learning. *Proceedings of the 12th International Academy for Information Management Annual Conference*. Atlanta, GA. (ERIC Document Reproduction Service No. ED422933)

Sutherland, T., & Bonwell, C. (Eds.). (1996). *Using active learning in college classes: A range of options for faculty*. San Francisco, CA: Jossey-Bass.

Tang, H-L., Lee, S., & Koh, S. (2000-2001, Winter). Educational gaps as perceived by IS educators: A survey of knowledge and skill requirements. *Journal of Computer Information Systems, 41* (2), 76-84.

Van Slyke, G, Trimmer, K., & Kittner, M. (1997, December). Integrating teamwork into information systems courses. *Proceedings of the 12th International Academy for Information Management Annual Conference*. Atlanta, GA. (ERIC Document Reproduction Service No. ED422950)

Biography

Sandra Poindexter is a professor of Computer Information Systems at Northern Michigan University, an institution recognized with a Laureate Award in the 2003 Computerworld Honors Program. She assists with the university laptop initiative and researches the adoption of Internet, technology, and other innovations in higher education. Professor Poindexter has published articles and presented papers at national and international conferences on all these topics, and is the author of a several textbooks on Internet browsers. As member of the university Teaching & Learning Advisory Committee, she works to promote excellence in teaching