AC 2012-5072: LEARNING THROUGH COGNITIVE DISSONANCE: ENGINEERING STUDENTS USE OF "PSEUDO PEER DIAGRAMS"

Ms. Sensen Li, Purdue University, West Lafayette

Sensen Li is a Ph.D. student in the engineering education program at Purdue University. She holds a M.S.Ed. in educational technology from Purdue University.

Dr. Sean P. Brophy, Purdue University, West Lafayette

Learning through Cognitive Dissonance: Engineering Students Use of "Pseudo Peer Diagrams"

Abstract

As part of an ongoing study of formative feedback systems we were exploring the potential of using "pseudo peer diagrams" as a low cost, indirect formative feedback method. We anticipated that learners with high motivation and academic talent usually take a central and active role to reduce cognitive dissonance by actively monitoring and regulating their thinking. Therefore, we used indirect formative feedback to diagnose learners' common errors associated with generating visual representations of problems they were solving or designing. This study asked first-year engineering honors students to generate flow charts to solve algorithm problems. We generated examples of possible solutions by combining common methods used by prior students. We called these diagrams "pseudo peer diagrams". To stimulate classroom discussion, we asked students to compare their diagrams with these pseudo peer diagrams. These classroom discussions could foster their self-check strategies which reduce instructors' need to process large amounts of information in real time and still provide effective real time feedback. This project replicated prior work of formative assessment used in technical fields like physics and engineering. Unique to this paper is the micro analysis of students' works. This micro analysis highlights the cognitive levels students engage in this kind of classroom based actively along with pedagogical content knowledge instructors need to effectively use this formative assessment in their classrooms. The results will illustrate a method for quickly classifying students' errors associated with evaluating engineering systems and recommendations for how to design formative feedback for classroom and individual learning system.

Background

Black & Wiliam conducted an extensive research review of more than 250 journal articles and book chapters on the effectiveness of formative assessment [1]. They proposed that efforts to strengthen formative assessment produce significant learning gains, and eventually raise academic standards in classrooms. Specifically, they pointed out that effective formative assessment involves collecting evidence about how students make progress during learning and making necessary instructional adjustment to facilitate their progress [2]. Heritage further built on Black and Wiliam's statements, and emphasized that formative assessment should be regarded as a process rather than a particular "test" [3]. As long as the assessment is specifically intended to generate feedback on performance to improve learning, it is formative [4]. Therefore, understanding how students think in the learning process, especially how they respond to feedback, increases the effectiveness of formative assessment.

While thinking about Black and Wiliam's criteria for effective formative assessment, it is instructive to look at how formative assessment is treated in the context of engineering education. A pressing goal of engineering education today is to acquiring knowledge to perform various actions, like design, troubleshooting and analysis [5]. In order to satisfy this goal, we need effective assessments capable providing details about how students cognitively approach engineering concepts and systematically comprehend problems. However, traditional approaches to engineering education are limited in their potential for formative assessment [6]. That is, even well designed lectures that use rhetorical question asking and well articulated explanations and demonstration lack feedback on students' comprehension. The introduction of technology to pull students' responses to conceptual questions is a great start to improving the effectiveness of formative assessment. It can be abstracted and described as 'sender–receiver' model which fails to highlight students' reasoning process [7].

This paper aims at facilitating students' comprehension of algorithm problems by asking students to generate diagrams and then providing pseudo peer diagram (PPD) as an indirect formative assessment strategy. The design and implementation of PPDs align with the nature of formative assessment, i.e., interaction between external feedback provided by instructors and internal interpretations produced by individual learners.

Theory

Research on using technologies to support the implementation of formative assessment in classroom has been widely conducted. Mazur used a peer instruction method to foster students' high-level reasoning as one way to provide formative assessment [8]. His system involved teacher questions posed to a large audience of participants designed to focus on specific concepts and multiple steps to solve. As students worked on the problem he listened to them explain to their peers how they approached the problem. He aggregated his observations to construct a solution he could share with the class and engage in a dialog about that solution. His aggregation of solutions were filtered by his specific learning outcomes, what he knew about common student difficulties and past successes in mediating these learning difficulties. This kind of knowledge of the pedagogy is important to maximizing the impact of instruction for all the students. His work has led to the design of effective teacher questions as the guidance to collecting meaningful and directed feedback from students. Similar to Mazur's Peer Instruction, Dufresne et al. developed a technology mediated personal response system called *Classtalk* [9]. This system facilitated presentation of discussion questions, collected students' responses and displayed these responses in large-scale lectures. They reported that students were engaged in active learning, and the overall classroom communication was greatly enhanced [9]. Since then many other versions of technologies have emerge to facilitate these kinds of formative assessment methods in the classroom. Some require low cost wireless transmitters that students could use to enter multiple choice and potentially short answer questions. Others use laptop technologies that allow students to share diagrams and large text expositories (e.g. siliconchalk, dynknow, and classroom presenter). These systems require real time response of students work product to generate an aggregated report of the classes' current understanding.

It is evident to see potential benefits to integrate personal response systems in classrooms, especially in large-scale lectures [6]. Based on students' immediate responses, instructors can decide to alter their instructional sequence to better meet the needs of the classroom. However, there are several challenges to implement dynamic formative assessment in engineering learning settings. For instance, much of the engineering learning and knowledge is represented in diagrams. This makes formative assessments a harder challenge to implement because it is difficult to gather students' results and to dynamically assess students' answers. This study attempted to overcome this challenge by presenting a pre-designed flow diagram and relying on self-check strategies and student-student interactions. Different from previous personal response systems, our study did not share a particular student's diagram in public, but presented a "pseudo peer" diagram designed by the instructor. The use of "pseudo peer" can not only cover a comprehensive version of knowledge that instructors expect students to learn, but also avoid students' uneasiness caused by the work being criticized and judged by peers. In addition, this pilot study informed us of the rationale and dynamics of incorporating personal response system in classroom settings, and paved the path for utilizing *Classroom Presenter 3* in our future study.

As engineering instructors, we usually see two major challenges existed when first-year engineering students learn programming. First, how to think computationally and design logic in the form of flow diagram; second, how to mechanically write syntactically correct code. We realize the abstraction of logic is the key to successful coding. Typically students rush to code directly without comprehending the logic. Therefore, they lack a clear definition of the problem they are trying to solve and a plan of action for how to solve the problem. As one instructional method, we ask students to generate diagram of their logic. Then, we introduce pseudo peer diagrams to reinforce the construction of visual representations as a roadmap to coding. We conjecture that pseudo peer diagrams are an effective tool to foster students' self-check strategy which reduces instructors' need to process large amount information generated by students in real time and still provide real time feedback that is tailored to common errors students demonstrate relative to specific concepts. We expected students to develop not only the fact knowledge, but also strategic skills for approaching and solving problems, and validate their results. Therefore, we need to train students to initially perform a qualitative analysis to identify the applicable principles and procedures before they attend to the details of applying these principles and procedures[9].Our research questions include:

1. Do pseudo peer diagrams promote students' understanding of how to generate a flow diagram?
2. Do pseudo peer diagrams enhance students' ability to evaluate and refine their programming logic?
3. What are students' attitudes towards the use of pseudo peer diagrams as a formative feedback?

Methodology

In order to provide a complete understanding of the research questions and explain quantitative results in detail, we utilize mixed-methods sequential explanatory design for this study [10]. Specifically, we quantitatively evaluate students' performance before and after presenting pseudo peer diagrams, and then qualitatively explore students' feedback on using pseudo peer diagrams to verify and augment the statistical results. The quantitative and the qualitative research were each designed to answering distinct research questions.

Participants

The participants in this study included seventy-four students enrolled in an honors version of the first year engineering (FYE) course at a large middle west university. These students self-selected into the course and were accepted on a first come bases. These students had a strong academic background and historically were highly motivated to achieve academically.

Procedure

The following problem was presented in the second exam of the semester. Before the exam, students received one week of instruction on generating flow diagrams and generated diagrams as part of their homework to develop Python code. For the problem, students were asked to generate a flow diagram to qualitatively analyze the problem and to abstract and interpret the logic before coding.

> Problem - We need to set up a single timer that displays "minutes: seconds" in python. Python has a library of time functions, including a time.time( ), which return the seconds as a floating point number. Therefore, the simple code expression that will wait for 1 second includes:
>
> ```
> startTime = time.time ( )
> While time.time ( ) – startTime < 1
>      Continue
> ```
>
> **USING THE SEPARATE SHEET OF PAPER PROVIDED**, construct a flow-diagram for an algorithm that will update the computer screen with current time elapsed from the start of the program. The display should resemble:
>
> Minutes: Seconds
>
> 00 : 00
>
> The program should be designed to end after 1 hour.
>
> b). (9 points) **AFTER** completing problem a), write a python script based on the flow-diagram you constructed for part (a). Points will not be given for this problem unless an attempt to plan an algorithmic solution as described above. *Once you begin coding you may correct your flow-diagram as necessary.*
>
> When you are finished, save your file as exam2_q12_login.py.
>
> Submit your file, exam2_q12_login.py, along with all other required exam files, to the appropriate location on Blackboard-Vista at the end of the exam period.

One week after the exam, an exam review session was administrated to students. Before the review session, the instructor designed a pseudo peer diagram (Figure 1), which was composed of essential elements and flows that we expect students could recognize. We also intended this pseudo peer diagram to serve as discussion starter for students to compare and contrast this diagram with their own work and inform instructors about their thinking. Therefore, we granted autonomy to students in the review session by asking students to self-check the pseudo peer diagrams without instructor's explanations and guidance.

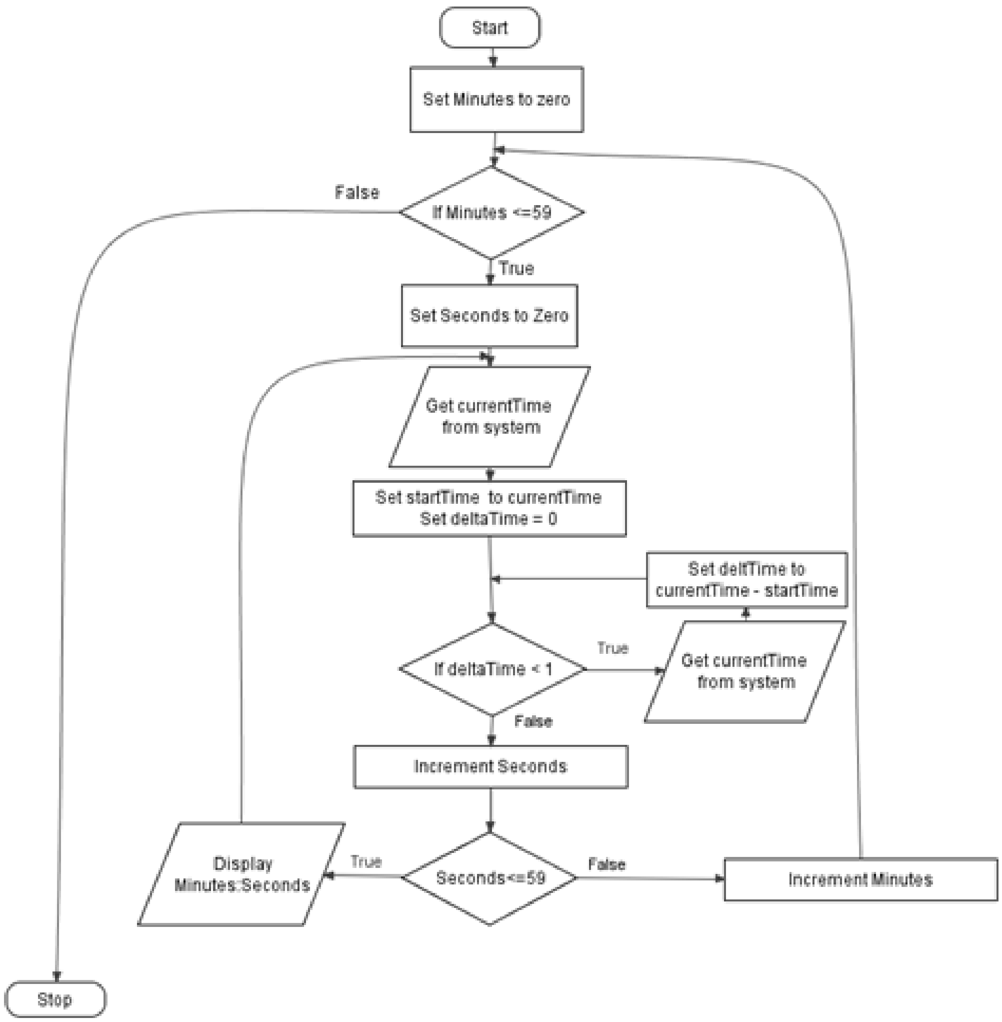


Figure One. Flow diagram of Algorithm

The following instruction was provided along with the pseudo peer diagram:

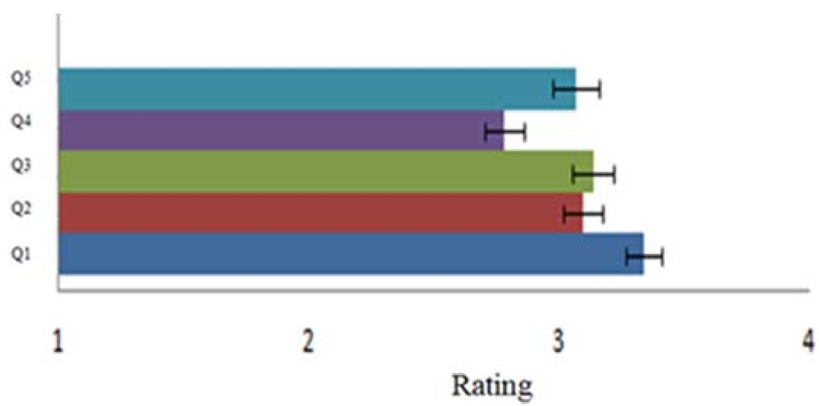
> Using a pen, refine your flow diagram to update the form of your flow diagram, and logic you missed in your flow diagram. If a large portion of your diagram is missing, then draw it to the side and indicate how it would fit into your initial diagram.

Students were also asked to fill in the survey (Appendix A) after they reviewed the pseudo peer diagram. The four-point Likert survey was aimed at understanding how students responded to the newly introduced instructional method. We removed the neutral options in the survey because often when students were unsure about the answers, they usually preferred neutral options [11]. It has been shown that whether removing the neutral option or not has little impact to the overall difference between a 4-point and 5-point Likert scale[12]. An open-ended question was incorporated at the end of the survey, asking about students' suggestions for the improvement of using pseudo peer diagram.

Results

We were interested in knowing if students made any improvement on their flow diagrams after they reviewed the pseudo peer diagram. Therefore, we used the same rubric (Appendix B) to evaluate students' initial and revised flow diagrams. Our unit of analysis in the rubric was threefold. First, we examined if students indicated any necessary changes on the *form* of the flow diagram, e.g. use of arrows to indicate directed flows. This demonstrated their knowledge of how to construct the flow diagram, or tool knowledge. Second, we examined if students *revised* their logic by incorporating critical logic components (nested loop in this problem). This knowledge is associated with algorithmic logic, or programming knowledge. Third, we examined if students *included the output* which displays the time. This is programming input and output knowledge.

A Wilcoxon Signed Ranks Test showed that pseudo peer diagrams elicit a statistically significant change in students' understanding of tool knowledge ($Z = -2.636$, $p = 0.008$), algorithmic logic ($Z = -5.915$, $p = 0.000$), and programming output ($Z = -2.000$, $p = .046$). Specifically, after reviewing the pseudo peer diagram, thirty-three students identified a change should be made by marking the error area; nineteen students merged their diagram with the pseudo peer diagram; and eighteen students revised their own diagrams by reflecting on the pseudo peer diagram.



(Rating scale: 1=strongly disagree, 2=disagree, 3=agree, 4=strongly agree)

Q1. System analysis
Q2. Flowchart construction procedure
Q3. Misses components
Q4. Confidence for complex systems analysis
Q5. Effective feedback

Figure 2. Student survey response to the use of pseudo peer diagrams in classroom

As illustrated in Figure 2, most students acknowledged the value of the pseudo peer diagram implemented in lecture. They agreed that pseudo peer diagrams facilitated their noticing of initial ideas of the system (Question 1), construction of flow diagrams (Question 2), and identification of missed components of the system (Question 3). Students were not very confident about their abilities to approach the complex design challenges even after reviewing the pseudo peer diagrams (Question 4), but they believed pseudo peer diagrams was an effective avenue for meaningful formative feedback. Question 1, 2 and 4 were selected to examine the internal consistency of the survey since all three questions were about students' self-reported interpretations of how pseudo peer diagrams facilitated system design and analysis. In addition, Cronbach's alpha is 0.706 (N=3), which indicates a high level of internal consistency for our scale with this sample.

In order to understand students' expectations about the application of pseudo peer diagrams, we also analyzed the open-ended survey question by means of a qualitative and inductive method in which categories of responses emerged from the data themselves, rather than being pre-determined and in which patterns were uncovered and made explicit[13]. Thirty three students (46%) indicated they required a different types of scaffoldings: 13 students (18%) preferred instructors or TA walking through the problem step by step; 7 students (9%) asked for more discussion in class or with team; 6 students (8%) requested more exercises of similar challenges; 5 students (7%) admitted that this was a difficult problem and they were not able to finish it on time; and 2 students (3%) mentioned that it was necessary to clarify the expectations and rules of constructing flow diagrams. The results led us to consider how to accommodate students' various needs in an effective and efficient way since we could not only target at one population and provided monotonous scaffolding to students. Our data also indicated some common attributes novice programmers demonstrate. Nine students (12%) argued that generating flow diagrams was not helpful to programming. As one student mentioned, "*I simply feel that flow diagrams do not help me at all, and I think that cutting straight to the programming would be a much more effective and hands-on approach to learning the material and skills.*" We believe comments like this result from students' focus on constructing syntactically correct code and underestimate the need for flow diagrams when the logic is trivial. Learning to construct syntactically correct code is accelerated by the immediate feedback provide by a computer compiler or interpreter. Debugging logic is more involved. As instructors, it is essential to point out that constructing visual representations develops students' skills to transform their logic into programming constructs, and helps students use the diagrams to communicate with others. Another misconception that we recognized in our data was 7 students (9%) complained that providing the single pseudo peer diagram narrowed their thinking and limited their innovation because there was more than one solution to this problem. Although we agreed providing multiple examples and insight is an illuminated strategy, for this particular programming challenge, the solution was not as open-ended an algorithmic task as the students suggest. Further we are presenting students with common and consistent programming constructs to assist in their design of logic. By limiting this set of constructs to sequential, conditional and repetition structures', we believe we help students develop consistent and efficient methods for constructing algorithms.

Discussion

Our quantitative results provided positive answers to the first and the second research questions, i.e. “do pseudo peer diagram promote students’ understanding of how to generate flow diagram?” and “do pseudo peer diagrams enhance students’ ability to evaluate and refine their programming logic?” We saw significant difference on students’ performance before and after reviewing PPDs as an indirect formative assessment strategy. PPDs helped students notice key features of diagramming a flow diagram which they initially missed. Furthermore, by comparing their diagram with PPDs, the majority of students identified the incorrect logic existed in their original work, and they made meaningful attempts to improve and refine their initial diagrams. To answer the third research question, our survey results indicated that most students had positive attitudes towards the use of PPDs as an effective formative feedback method for their learning. Yet some students still required more scaffolding beyond reviewing PPDs, such as direct feedback from instructors or TAs, class or team discussion and clear goals and criteria. This could potentially indicate that some students were still not familiar enough with the rules for constructing the diagrams and practice translating their logic into a flow diagram. This could be tested by providing students with additional flowcharting constructions activities with similar feedback methods.

Our qualitative results further exposed that some students preferred coding directly rather than generating diagrams to analyze their logic first. Our objective was to help students realize the importance of using these diagrams to plan their design of a complex algorithm before actually constructing a program. The results from this study showed that every student made logic changes to their diagram as part of the pseudo peer diagram comparison. Therefore, the activity of generating the diagram and reflecting on it was a productive task for all students. We needed to help students see that it was more effective discovering their errors early using the diagram compared to a cycle of generate, compile, run, debug, and rewrite process. It is critical for students to realize the value to decompose the problem through further analysis and sense making before producing code.

Future Work

This study illustrates the pedagogical affordances of asking students to compare and contrast their work with a model illustrating a potential solution. This research illustrates that some students are ready to evaluate their own work and have sufficient tool and programming logic knowledge to self correct their work. Still others need additional assistance in constructing and evaluating their work. They may also need explicit feedback to notice their errors. We believe they also need additional lessons targeting their conceptual understanding to further enhance conceptual understanding [14]. Therefore, we are working on technologies that students can construct diagrams and then receive varying levels of feedback to support their conceptual understanding. These kinds of practice environments can produce the tool and programming knowledge needed to construct flow diagrams. Further work of the students on more complex problems will most likely be needed to increase students’ proficiency with these construction skills and to develop the abstract conceptual understanding they will need to manage complex tasks.

These studies provide general knowledge about students' approach and performance on constructing graphical representations of their knowledge. We seek to have a more comprehensive explanation of how students approach these problems. We are planning to use think aloud protocols with a representative sample of the various skill levels of these students. Through these interviews we will be able to provide a richer description of how students approach these problems and the kinds of challenges they experience as they work on these problems.

Finally, technology can provide an even stronger formative assessment experience in the classrooms. As mentioned earlier, classroom response systems can be used to poll students thinking with leading questions. These challenge students' conceptual understanding using multiple choice questions or short answer. This has proven to be quite effective. Engineering activities require the use of graphical representations as well. New systems, like Dyknow, SiliconChalk and Classroom presenter, provide students with a mechanism to draw representations with computer tools, and then share them with an instructor. These are sent to instructors who can then view the submission and share students' product with the class. We are working on a study to use the Classroom Presenter system as a method for investigating the potential of these systems for engineering education learning environments.

Conclusion

In our research we treat formative assessment as a powerful engine for teaching and learning, not as "a more frequent, finer-grained test" [3]. In addition, as engineering education researchers, we are interested in graphical representations of knowledge because they are fundamental to engineering thinking. Therefore, our focus is on how to deliver meaningful information in an effective way using visual as well as linguistic information. Pseudo peer diagram is one attempt we made to facilitate the delivery and elicit students' thinking on logic before coding. The results indicate students can generate and initial visual representation that transforms a work problem into graphical logic. Further we found that students have various levels in their ability to generate and evaluate these images. This suggests that some students are ready to use this kind of formative assessment method, but other students may not be completely ready to perform this task on their own. Therefore, multiple methods are necessary to assist in this approach. We are working to define these methods and explain how these methods can be achieved.

This initial study helped us identify the details of instructional design and classroom implementation, informed us about the diverse types of scaffolding students required, and paved the path for our future studies around how to design automated personal response systems involving engineering activities (e.g. algorithm design, computational modeling for analysis) which use visual representations. We may expect more discussion and studies on the theoretical underpinning of pseudo peer diagrams, as well as the use of Classroom Presenter 3 in our future research.

Acknowledgements

This research was made possible with a grant from the National Science Foundation (Award Number – DUE-0817486)

Bibliography

[1] Black, P., & Wiliam, D. (1998a). Assessment and classroom learning. *Assessment in Education: Principles Policy and Practice, 5*(1), 7–73.
[2] Black, P., & Wiliam, D. (1998b). Inside the black box: Raising standards through classroom assessment. Phi *Delta Kappan, 80*(2), 139-147.
[3] Heritage, M. (2010). *Formative assessment and next-generation assessment systems: Are we losing an opportunity?* A project of Margaret Heritage and the Council of Chief State School Officers (Paper prepared for the Council of Chief State School Officers). Washington, DC: Council of Chief State School Officers.
[4] Sadler, D. R. (1998). Formative assessment: revisiting the territory. *Assessment in Education, 5*(1), 77–84.
[5] Brophy, S. P., Klein, S., Portsmore, M., & Rogers, C. (2008). Advancing engineering education in the P-12 classrooms. *Journal of Engineering Education 97*(3), 369–387.
[6] Roselli, R. J., & Brophy, S. P. (2006). Experiences with formative assessment in engineering classrooms. *Journal of Engineering Education, 95*(4), 325-333.
[7] Biesta, G .(2004). *Mind the gap! Communication and the educational relation.* In Bingham, C., & Sidorkin, A .eds. No Education without relation. New York: Peter Lang.
[8]Mazur, E. (1997). *Peer Instruction: A user's manual.* Englewood Cliffs, NJ: Prentice Hall.
[9] Dufresne, R. J., Gerace, W. J., Leonard, W. J., Mestre, J. P., & Wenk, L. (1996). Classtalk: A classroom communication system for active learning. *Journal of Computing in Higher Education,* 7, 3-47.
[10] Creswell, J. W., & Plano Clark, V. L. (2007). *Designing and conducting mixed methods research.* Thousand Oaks, CA: Sage.
[11] Presser, S. & Schuman, H. (1980). The Measurement of a Middle Position in Attitude Surveys. *Public Opinion Quarterly, 44*(1), 70-85.
[12] Armstrong, R. (1987). The midpoint on a Five-Point Likert-Type Scale. *Perceptual and Motor Skills,* 64, 359-362.
[13] Patton, M. (2002). *Qualitative research and evaluation methods.* 3rd ed. Thousand Oaks, CA: Sage Publications.
[14] Hunt, E. & Minstrell, J. (2004). *The DIAGNOSER Project: Formative Assessment in the Service of Learning.* Paper presented at the International Association for Educational Assessment, Philadelphia.

Appendix one. students' responses towards the implementation of pseudo peer diagrams

Instructions: As part of our discussions in class we often ask you to generate an answer to a multiple step questions, such as constructing a flow diagram, draw a functional block diagram, and generate lists of specifications. Then, we provide a potential solution for you to compare with your solutions. Please respond to these statements in a way that best describes your experience with this approach.

1. Comparing my diagrams with a potential solution helps me notice important ideas about a system or the problem definition that I missed in my initial solution.

- o Strongly Agree
- o Agree
- o Disagree
- o Strongly Disagree

2. Comparing my diagrams with a potential solution helps me notice important things I missed on how to construct a proper diagram (e.g. a functional block diagram, flow diagram).

- o Strongly Agree
- o Agree
- o Disagree
- o Strongly Disagree

3. Listening to my peers discussion of the potential solution increases my ability to better understand what I missed in my solution.

- o Strongly Agree
- o Agree
- o Disagree
- o Strongly Disagree

4. I find I am more critical of how I approach complex problems like the design challenges and term projects after doing exercises like comparing my diagrams with a potential solution.

- o Strongly Agree
- o Agree
- o Disagree
- o Strongly Disagree

5. Comparing my diagram with a potential solution followed by class discussion provides more effective feedback than receiving a corrected homework assignment.

- o Strongly Agree
- o Agree
- o Disagree
- o Strongly Disagree

6. Please provide comments and suggestion you think would improve the benefit of these kinds of activities to your learning more about how to apply engineering ideas and properly use tools.

Appendix two. Rubrics used for flow diagrams.

Categories	Content	Weight
Form (Tool knowledge)	Use of “start /end” in the flow diagram. Use of “top/down” or “left/right” flow in the flow diagram. Use of arrows to represent the direction. Use of proper symbols. Use of “true/false” or “yes/no” paths for conditional tests.	1 1 1 1 1
Logic (Domain knowledge)	A. Loop of minute: Set minutes to zero; If minutes <=59; Loop B and C; Increment minutes; B. Loop of second: Set seconds to zero; If seconds <= 59; Loop C; Increment seconds; C. Waiting loop. Set startTime to currentTime. Set deltaTime = 0; If deltaTime < 1; Get currentTime from system; Set deltaTime to currentTime-startTime.	 1 1 1 1 1 1 1 1 1 1 1 1
	Programming output: Display minutes : seconds / Display minute:00	1