

Things are Clicking in CS4

Roy P. Pargas
Clemson University
Department of Computer Science
Clemson, SC 29634-0974
864.656.5855
pargas@cs.clemson.edu

Dhaval M. Shah
Clemson University
Department of Computer Science
Clemson, SC 29634-0974
864.656.1362
dshah@cs.clemson.edu

ABSTRACT

This paper presents and discusses a *modified approach* to teaching an algorithms and data structures course (CS4). The approach relies on frequent evaluation of student understanding of course content and enables the instructor to experiment with various exercises to facilitate peer-instruction and cooperative learning. It uses a web application called *MessageGrid* which enables instructor and students, each of whom has a laptop computer with wireless access to the web, to interact in a variety of ways both in and out of class.

Categories and Subject Descriptors

K.3.1 [Computers and Education]: Computer Uses in Education – Collaborative Learning

General Terms

Experimentation

Keywords

Peer-instruction, collaborative learning, assessment, laptop course

1. INTRODUCTION

Peer-instruction and cooperative-learning have been used in teaching for at least the past two decades. Mazur [1] has been developing and implementing variations of peer-instruction techniques for calculus- and algebra-based introductory physics courses for non-majors at Harvard University. Prey [2] accurately states that there is a mismatch between what is taught in computer science and how it is taught, versus what the students need to succeed in their computer science careers. She discusses the cooperative learning approach implemented in closed laboratory exercises in the first four core courses of the computer science curriculum at the University of Virginia. In 1996 and 1997, Wills [3] led NSF-sponsored workshops that focused on peer-learning in freshman and sophomore computer science courses. The outcomes included specific cooperative projects and tasks for use

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM SIGCSE '06, March 1-5, 2006, Houston, TX, USA.
Copyright 2004 ACM 1-58113-000-0/00/0004...\$5.00.

in introductory computer science curriculum and the means for evaluating and assessing these tasks.

This paper presents and discusses a *modified approach* to teaching an algorithms and data structures course (CS4). The approach relies on frequent evaluation of student understanding of course content and enables the instructor to experiment with various exercises to facilitate peer-instruction and cooperative learning. The course, called CPSC 212, serves primarily freshman and sophomore computer science majors. Each has a laptop computer with wireless access to the Internet.

A similar style of teaching has been used effectively at Clemson in large chemistry classes (90-250 students) using audience response polling devices [4]. We have adopted the practice in CPSC 212 using custom-designed software and the students' laptops. Each week, the instructor, two graduate teaching assistants and fifty undergraduate students meet for two 75-minute lecture periods and one 110-minute closed lab. The instructor leads the lecture sessions and the teaching assistants alternate leading the labs.

The modification in the teaching style involves the manner in which course material is *delivered* during each lecture period. Section 2 describes a web-based tool called *MessageGrid* which plays a major role in the delivery of the material. Section 3 describes a typical lecture period involving real-time assessment of student understanding of the day's lecture topic. The assessment and resulting feedback is made possible by a feature of *MessageGrid* called *clickers*, also described in Section 2. Section 4 discusses the results of three sample questions presented to CPSC 212 students using clickers. The discussion shows how clicker feedback can provide valuable information to assist the instructor in delivering content. Section 5 provides student assessment of *MessageGrid* and clickers. Section 6 offers conclusions and plans for future work.

2. MESSAGEGRID

MessageGrid [5,6] is a web-based software system that allows the instructor and students to interact electronically during (as well as outside of) class. By designing a two-dimensional grid with user-specified rows and columns, the instructor tailors the grid to support and complement the goals of the day's lecture. Students can post one or more messages¹ in a grid cell. *MessageGrid* can be used to view short student programs, collect and grade homework, solicit questions before tests, and provide feedback to

¹ A message may be text, an image, a document, a spreadsheet, an audio or video file, a PowerPoint presentation, a student-designed html page, indeed anything that can be downloaded or displayed by a browser.

students, initiate class discussions, elicit opinions on a topic, or preview student documents.

This paper highlights the use of a *MessageGrid* feature called *clickers*. The term *clicker* is used to mean a *single-choice, multiple-option question*, displayed to the students as a radio button question. The instructor develops a clicker within *MessageGrid* and releases it to the students. As soon as students answer the clicker, results are tabulated. When released by the instructor, these results may be viewed by the students in the form of frequency tables and histograms. Anyone who remembers the popular television show “How to Become a Millionaire” will recognize a *MessageGrid* clicker as a web version of the “lifeline question” thrown by the contestant to the audience.

The instructor controls when to release the clicker question and when to release student responses. Students can only see summary results; the instructor can view details, such as which answer option was selected by each student, when the answer was submitted, and which students did *not* submit an answer. The instructor may *freeze* a clicker preventing further submissions. Questions may be developed ahead of time or may be generated on-the-fly. In brief, clickers can provide *instant* feedback to both instructor and students providing clues as to whether the course material is or is not being understood.

3. MODIFIED LECTURE

A university mandate requires all entering students at Clemson University to arrive with laptop computers with wireless access to the web. Many instructors have sought to modify their traditional lectures into ones that include active participation by the students using their computers [7-10]. For these instructors, *MessageGrid* is a tool that allows them to interact with their students using laptop computers before, during or after lecture. This section describes changes being attempted in CPSC 212 lecture.

3.1 Pre-Lecture Reading Assignment

Students are assigned sections of the text to read before each class period. Each day a grid is prepared with student names for rows and two columns labeled “Questions” and “Response”. An instructor’s view of a sample grid is shown in Figure 1.

Ask a specific question about a topic you do not understand in Sections 5.5 - 5.7. Your question should not be vague as in “I do not understand analysis of algorithms.” Your question should make it clear exactly what you do not understand about complexity. If you understand everything in the assigned reading, then enter “I understand everything!” Deadline: 11:59 pm, Monday, September 5.

CPSC 212, SIGCSE	Questions (Sections 7.5-7.7)	Response
██████████	<p>██████████ (9/5/2005 8:50:47 PM) (edit) (delete) (reply)</p> <p>I have a question regarding the differences in big-Oh, big-Omega, big-Theta, and little-Oh.</p> <p>(post)</p>	(post)
██████████	<p>██████████ (9/5/2005 9:00:39 PM) (edit) (delete) (reply)</p> <p>How uniformly distributed does data have to be in order to properly use an interpolation search? The books example of {1,2,4,8,16...}</p> <p>(post)</p>	(post)
██████████	<p>██████████ (9/5/2005 8:55:58 PM) (edit) (delete) (reply)</p> <p>I understand everything!</p> <p>(post)</p>	(post)

Figure 1. Sample reading assignment grid

The day before the lecture, students are asked to post a *specific* question about anything that they do not understand in the reading assignment. Students who have no questions are asked to post “*I understand everything!*”

This grid assignment helps both the instructor and the students in several ways. It forces the students to at least *acknowledge* the reading assignment and to post one or more questions about it on a grid, or claim that they understand the material completely. Approximately two-thirds of the students regularly claim that they have read the material and understand it completely, far too many to be believable. The remaining one-third, however, *do* post good questions which provide valuable clues to the instructor as to which specific topics are most difficult. In practice, this proves to be quite valuable in fine-tuning the next day’s lecture. The instructor may respond to one or more of the questions and make the grid available to the entire class providing review material for the next test.

3.2 Submission of Solutions to Problems

Grids are also used in CPSC 212 to get students to work on short exercises (e.g., “Show the contents of a hash-table after storing a set of values using hash function $f_1(x)$ and employing double-hashing with function $f_2(x)$ to resolve collisions.”). We give such exercises sometimes before, sometimes during, and sometimes after a lecture on the topic. Submissions can be made public (viewable by the entire class) or private (students see only their own submissions). Also submissions may or may not be anonymous. All of these *MessageGrid* options are controlled by the instructor who can therefore custom-design a grid to suit different teaching situations and meet different teaching goals.

In CPSC 212, we frequently conduct in-class exercises after a brief (15- or 20-minute) lecture on the topic. We allow students either to work individually or to collaborate in small groups, but we require that each student post an individual response to the assigned exercise. The rationale is that although collaboration and peer-instruction can enable two students to work toward the solution of the problem, or can help a weaker student learn from a stronger student about how to approach a problem, ultimately *every* student must be able to correctly compose and post a response.

These low-stakes exercises are an excellent way to conduct frequent evaluations of student abilities. We can ask for code snippets, short proofs, two or three sentence explanations, real-life use of the data structure of the day, short answers, and other similar exercises. Responses can be organized by name on a single webpage, making it easy to scan student submissions visually and spot potential problems. The instructor can post solutions to the problems as well, so that the students themselves can determine if they answered a question incorrectly. And because credit is given for simply submitting a solution (whether correct or incorrect), grading is not a problem.

MessageGrid can also be used to review preliminary versions of student documents, for example, draft versions of PowerPoint presentations, project proposals, or algorithm designs. If the grid is public, comments on the submissions may be posted by the instructor and thus benefit *all* students. It is also possible for the instructor to have students submit peer-evaluations of their classmates’ work. The instructor can then critique the peer-evaluations. The teaching opportunities are numerous!

3.3 Ask, and Ask Again

The *MessageGrid* clicker opens the door to real-time assessment of student learning and can provide valuable feedback to instructors. We have used clickers in several different ways, but we describe here the process that we have found to be the most productive and informative for both student *and* instructor.

The 75-minute lecture period is divided into several short segments: (a) instructor-led lecture and discussion, (b) question-and-answer using a clicker, (c) possible release of summary results, (d) peer-discussion and collaboration, (e) repeat question-and-answer using a second identical clicker, (f) release of results of both clickers, (g) instructor-led discussion.

In (a) the instructor introduces and briefly lectures on the topic for the day. This typically takes 15-20 minutes in a 75-minute lecture period. In (b) the instructor releases a pre-designed clicker to the students. Sometimes student answers are released immediately (c) revealing the table and histogram summarizing student responses. Students are *not* told the correct answer. They are, however, instructed to try to *convince* their neighbors (d) that the answer they submitted is correct, or *be convinced* that their neighbor's answer is correct. They may also agree to disagree. At the end of the discussion (usually no more than five minutes) the students answer (e) a second, identical clicker. When all have answered the second question, both sets of results are released (f) and the instructor leads the class through a post-mortem discussion. This discussion benefits *both* students and instructor. Students who missed the question realize that they are missing something and should do something about it. The instructor realizes that what that he thought was simple and obvious apparently isn't, and *he* should do something about it! There is usually enough time in a 75-minute period to repeat this process with a second topic.

4. THREE EXAMPLE CLICKERS

This section discusses the results of three examples of clicker use in CPSC 212. These questions were asked in the second week of the semester; students in the class had taken CS1 and CS2 or their equivalents and had some but not much experience with trees and recursion. The topic for the first clicker pair is *basic algorithm analysis*. After a lecture on the importance of analysis, the most frequently used orders of complexity, and how to count loop iterations, the students were presented with the first clicker.

Algorithm A	Algorithm B
<pre>int m = n; while (m > 0) { for (int i=0; i<n; i+=2) { // 2 multiplications // 3 additions } // i m /= 2; } // while</pre>	<pre>for (int i=0; i<n; i++) int m = 0; while (m <= n) { // 1 multiplication // 1 addition m += 10; } // while } // i</pre>

Figure 2. Two algorithms to be analyzed

Students were asked to analyze two algorithms (see Figure 2) and to select the best answer among: (1) A is $O(n \log n)$ and B is $O(n)$, (2) A is $O(\log n)$ and B is $O(n^2)$, (3) A is $O(n \log n)$ and B is $O(n^2)$, (4) A is $O(n)$ and B is $O(\log n)$, (5) A is $O(\log \log n)$ and B is $O(n^2)$, (6) None of the above. The results of the first clicker (Figure 3) were released to the students. Students were *not* told which answer was correct.

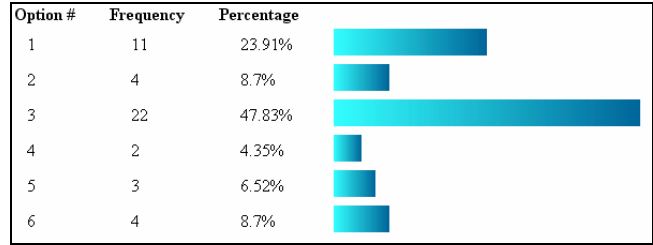


Figure 3. Algorithm analysis

The students were then instructed to try to convince their neighbors that their answers were correct. After a few minutes of discussion, students answered a second clicker, the results of which are shown in Figure 4.

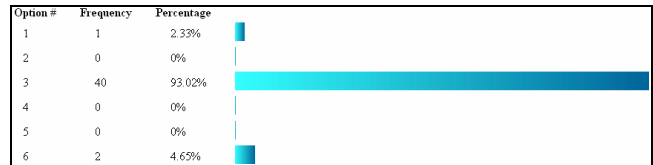


Figure 4. Algorithm analysis revisited

All but three students selected the correct answer. We found this to be quite remarkable, delightful, and humbling (since we had absolutely *no* input to the discussion between clicker responses). Even more important is the fact that *MessageGrid* records the identities of all respondents (available only to the instructor) and so the instructor is able to take the three outliers aside for private instruction on algorithm analysis.

Note that the results of the first clicker were shown to the students *before* the group discussion. Is it possible that the results (specifically that option (3) in Figure 3 received the largest number of responses) biased some students toward option (3) even before the discussion? To test this in example 2, the instructor did *not* show the results of the first clicker until after the second clicker was answered. The topic was Towers of Hanoi and after a *very brief* applet demonstration of the solution to the problem for sizes $n=3$ and $n=4$, the question posed was: *How many moves does it take to solve the general n-disk problem?* The options were: (1) 2^{n-1} , (2) $2^{n-1}-1$, (3) $2^{n-1}+1$, (4) 2^{n+1} , (5) $2^{n+1}-1$, (6) $2^{n+1}+1$, (7) 2^n , (8) 2^n-1 , (9) 2^n+1 , (10) None of the above. Figure 5 shows the results of the first clicker.

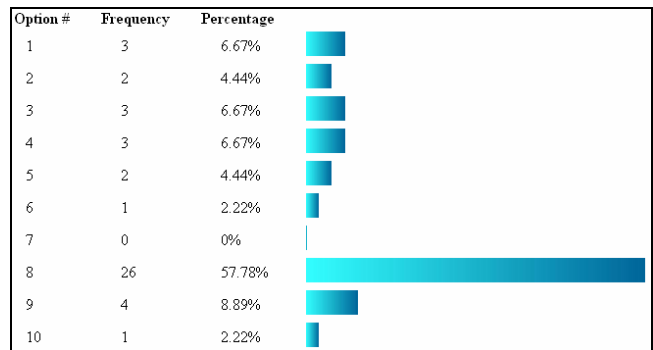


Figure 5. Towers of Hanoi

Figure 6 shows that even though the first clicker results were not revealed, after the discussion over 90% of the students selected the correct answer.

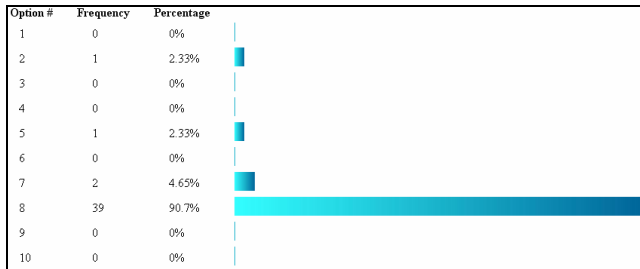


Figure 6. Towers of Hanoi revisited

Does this approach always work? Not always. The two questions above are relatively easy. A more difficult task for this group of beginning CPSC 212 students was identifying the behavior of recursive algorithms. In example 3, the question was: *What does the following recursive algorithm do?*

```
public static int Mystery(TreeNode t) {
    if (t == null)
        return 0;
    if ((t.left != null) || (t.right != null)) {
        return Mystery(t.left) + Mystery(t.right);
    }
    else
        return 1;
} // Mystery
```

The options were: (1) Counts the total number of nodes in the tree, (2) Counts the number of interior nodes in the tree, (3) Counts the number of nodes with one child in the tree, (4) Adds the values of all nodes in the tree, (5) Adds the values of all interior nodes in the tree, (6) Adds the values of all leaves in the tree, (7) None of the above. The algorithm (which counts the number of leaves of the tree) stumped all but three students (see Figure 7) who correctly selected option (7).

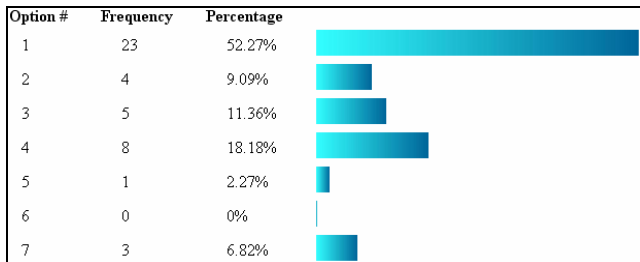


Figure 7. Mystery recursion

Even after discussion (Figure 8), the majority of students still selected option (1). Throughout this exercise, the instructor sensed a palpable uneasiness among the students. This prompted him to ask (after the second clicker was answered, but *before* revealing the results) “How many would like to change their answer? And if so, from what to what?” Four students said that they would have liked to change their answers from (1) to (7) and two said from (2) to (7). So it appears that the true results are a bit more *evenly* bipolar than depicted in Figure 8, although still leaning heavily toward the *wrong* answer.

The ensuing discussion was revealing. Although some students sensed that none of the first six answers was correct, they did not understand enough about recursion to correctly describe the behavior of the algorithm and, as a result, were not confident enough to select “None of the above”. When the instructor proceeded to explain the algorithm and how to analyze recursive algorithms in general, he sensed an uncommon and distinct

attentiveness in the class. He surmises that this attentiveness was due to the students’ sudden realization (i.e., three-fourths of the class had a rude awakening) that they did *not* understand recursion and that they had better pay attention! For the instructor, this was, to say the least, an extremely satisfying lecture.

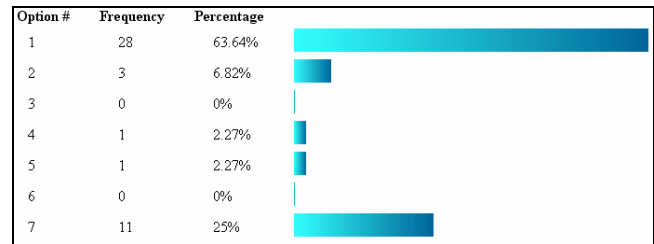


Figure 8. Mystery recursion revisited

When we say that this approach does not always work, as demonstrated by example 3, we mean that students will not always gravitate to the correct answer. The information obtained, however, is still quite valuable because it can point out a serious lack of understanding of a specific topic, as it did in this example. Using clickers, the instructor can gain significant understanding about the students’ strengths *and* weaknesses and can use this to decide what to focus on in subsequent lectures².

5. ASSESSMENT

We asked the students to assess *MessageGrid* and clickers by rating seven statements on a Likert scale with options: (a) Agree very strongly, (b) Agree strongly, (c) Agree, (d) Disagree, (e) Disagree strongly, (f) Disagree very strongly, (g) Not applicable. The statements and responses are summarized in Table 1.

The results of the survey are very encouraging. It appears that the students believe that *MessageGrid* clickers and discussion with neighbors help them learn and understand course material (statement 1: 96% agree, 3: 98% agree). Clicker feedback also appears to be very helpful (statement 6: 96% agree). Working alone during class exercises is effective with a number of students (statement 5: 7 out of 45 or 16% agree) suggesting that the instructor should always make available the option *not* to discuss with seatmates. Only 76% agree (statement 2: 35 out of 46) that posting questions on *MessageGrid* helps them recognize difficult sections in the reading and 7% admit that they do not participate at all in this activity. Over 2/3 of the class (statement 4: 30 out of 44) agree that discussion of course topics outside of class helps them understand course material well; 1/3 (13 out of 44) do not discuss course topics outside of class. Finally, 87% (statement 7: 40 out of 46) find *MessageGrid* easy to use.

6. CONCLUSIONS AND FUTURE WORK

We have only begun to use *MessageGrid* and clickers on a regular basis but already, we are gathering information about the strengths

² Because of the results shown in Figures 7 and 8, on the next lecture day after example 3 was given to the class, the instructor *changed* his lesson plans and spent the entire period discussing sample recursive algorithms on trees, how to write them, how to trace them, and how to analyze them. At the end of the period, two clickers produced 100% and one produced 94% correct results. Were it not for example 3, the instructor would have blithely moved on to the next course topic, totally oblivious of the clear lack of understanding among the students.

Student Assessment Results	(a)	(b)	(c)	(d)	(e)	(f)	(g)
1. Working with <i>MessageGrid</i> clickers helps me learn the course material	8	23	12	1	1	0	0
2. Posting questions related to the reading assignments on <i>MessageGrid</i> helps me recognize what is difficult in the reading	2	15	18	6	2	0	3
3. Discussing course topics with my seatmates in class helps me better understand the course material	15	20	10	1	0	0	0
4. Discussing course topics with my classmates <i>outside</i> of class helps me better understand the course material	10	9	11	0	1	0	13
5. Working alone during class is a more effective learning strategy for me than discussing topics with my classmates	0	2	5	19	9	11	0
6. The immediate feedback from clickers helps me to focus on weaknesses in my understanding of the course material	14	19	11	1	1	0	0
7. <i>MessageGrid</i> is an easy-to-use class collaboration tool	10	14	16	4	2	0	0

Table 1. Student assessment of MessageGrid and clickers

and weaknesses of this particular class very rapidly. The feedback about where students are having difficulty enables the instructor to adjust lesson plans for subsequent lectures. And because the feedback is obtained *long before* a major test, there is opportunity to remediate. We suspect that a student's lack of understanding of the material may, with careful use of *MessageGrid* clickers, be exposed early, before it has a chance to become major problem.

The very positive student evaluations encourage the authors to continue experimenting with *MessageGrid* and clickers in various teaching scenarios, maintaining a journal and documenting the results of each experiment. At the end of the semester, we plan to organize a short report detailing what techniques worked and what did not. We will also ask the students to take the evaluation survey again at the end of the semester and will compare results.

The future of *MessageGrid* software development is busy. Clemson faculty using *MessageGrid* meet regularly and provide a continually expanding list of suggestions for improvement. In the near future, we plan to add clicker templates, providing standard Likert [11] scales as answer options.

We are also designing alternative views for grid content. One professor is interested not only in *what* students post when they *reply* to a message, but also in *what order* they are posted. He has requested for a *tree-structured* display of grid content, where an original submission becomes the root of the tree, and replies are (recursively) its descendants. At the click of a button, therefore, the view of the content will switch from a grid to a *forest of trees*.

We are also working toward allowing tablet PC *ink* submissions. The goal is to enable a student who has a tablet PC or PDA to submit ink drawings of various data structures. In a class like CPSC 212, being able to draw representations of different data structures will be very convenient and helpful.

7. ACKNOWLEDGMENTS

We gratefully acknowledge support in the form of a multi-year Faculty Fellowship from the Clemson University ETS-OTEI Laptop Faculty Development Program and a 2003 Content and Curriculum Development Grant from Microsoft Corporation. We also wish to thank Jim Witte and Svetlana Drachova for their help in designing the evaluation survey and tabulating the results.

8. REFERENCES

- [1] Crouch, Catherine H. and Mazur, E., "Peer-Instruction: Ten years of experience and results", *American Journal of Physics*, 69(9), September 1997, pp. 970-977
- [2] Prey, J.C., "Cooperative learning in an undergraduate computer science curriculum", *Proceedings of the 1995 ASEE/IEEE Frontiers in Education Conference*, November 1995. pp. 3c2.11-3c2.14 vol.2
- [3] Wills, C.E., Cordes, D., Deremer, D., Klein, B.J., McCauley, R.A. and Null, L., "Application of peer learning to the introductory computer science curriculum", *SIGCSE Bulletin*, 29(1), March 1997, pp. 373-374. <http://www.cs.wpi.edu/~peercs>
- [4] Draper, S. "Electronically enhanced classroom interaction", <http://www.psy.gla.ac.uk/~steve/ilig/handsets.html>
- [5] Pargas, R.P., Levin, A.R. and Austin, J., "Work-In-Progress: Providing Interactivity in a Technology-Rich Classroom", *Proceedings of the 2005 ASEE/IEEE Frontiers in Education Conference*, Indianapolis, IN, October 19-22, 2005 (*to appear*)
- [6] Pargas, R.P., "Using *MessageGrid* to Promote Student Collaboration", *Cognition and Exploratory Learning in a Digital Age 2005*, Porto, Portugal, December 14-16, 2005, (*to appear*)
- [7] Pargas, R.P. and Weaver, K.A., "Laptops in Computer Science: Creating the 'Learning Studio'", *New Dimensions for Teaching and Learning*, Vol. 2005, No. 101, Spring 2005, L.B. Nilson and B.E. Weaver, eds., pp. 43-49.
- [8] Moss, W.F., Pargas, R.P., Grimes, L.W., and Weaver, B., "Technology + Innovation = Pedagogy", *Proceedings of the International Conference on Multimedia and Information and Communication Technologies in Education*, Badajoz, Spain, 2003, pp. 1065-1069
- [9] Campbell, A.B. and R.P. Pargas, R.P., "Laptops in the Classroom", *Proceedings of the ACM 34th SIGCSE Technical Symposium on Computer Science Education*, Reno, Nevada, 2003, pp. 98-102
- [10] Herron, J. and Pargas, R.P., Blending Technology into Review Sessions, *Proceedings of the 15th International Conference on College Teaching and Learning*, Jacksonville, FL, 2004, pp. 47-55
- [11] Siegle, D., "Likert scale", <http://www.gifted.uconn.edu/siegle/research/InstrumentReliabilityandValidity/Likert.html>