Experience With A Literate Approach to Computer Science

Douglas Thain[1] and Christian Poellabauer[2]

Abstract - **We have observed anecdotally that many beginning graduate students, while technically skilled, are deficient in the ability to communicate technical ideas orally and in writing. To address this, we have developed a literate approach to graduate education. This approach has two components: reflection on a series of technical readings appropriate to a course, and a highly literate course project where students develop and polish a paper through several milestones and evaluations. We have gained experience with this approach in four classes over two years. We observe that students do respond to the literate format, and are able to draw connections between the class readings and their own writing.**

INTRODUCTION

For many students, the transition from college to graduate school requires a dramatic change in perspective. Most college students are accustomed to absorbing well-defined morsels of knowledge and then answering focused technical questions about the knowledge. (Consider the daily readings followed by numbered questions in any college textbook.) However, graduate students must have a very different mode of learning: they are expected to ask unanswered questions, to explore new fields of knowledge, to connect related but distant pieces of information, and to explain methods and results of their own creation. To engage in these tasks, graduate students must have outstanding communication skills.

Of course, communication skills are hardly limited to graduate school. Those earning advanced degrees in science and engineering are not meant to become technicians in a field of study, but are expected to take leadership roles in business, academia, and government. For these graduates, their job will not be to write code, but to explain, motivate, organize, and persuade. Effective leaders in any field must combine technical knowledge with outstanding communication skills.

We have observed anecdotally that incoming graduate students are not well prepared for this role. Although recent college graduates have excellent technical skills, they have difficulty communicating about technical matters. In evaluating student papers, we often find that students describe their work in very focused technical language, but cannot explain the work in general terms, place the work in a broader context, or explain why it is useful to humanity in general. In evaluating student-given lectures, we often find that they leap right into technical results without first guiding the listener from a high-level problem down into the technical details of a solution. Without a good delivery, important technical results are likely to be misunderstood or ignored completely.

To address this, we are developing a "literate" course format designed to develop the ability of students to read, write, and speak effectively. The format has two complementary components. The first is continuous critical reading. Students read a technical paper for every class period. In addition to discussing the technical content, attention is also paid to discussing and evaluating the clarity and effectiveness of the writing. The second component is a technical but highly literate course project. Throughout the semester, students develop the literate aspects of their project alongside the technical aspects: they must write position papers, give formal talks, conduct peer reviews, and even practice a thirty-second "elevator" talk. The goal is to continually polish and refine ideas so that the final paper is a high quality result, not a last-minute rush job.

A work-in-process paper at Frontiers in Education 2005 described a one-semester pilot of this course format. Since then, we have evaluated four courses over two years with student surveys and instructor reflections. We offer a detailed description of the format and materials and describe student comments and instructor experiences. We find that students do respond to the literate format, and are able to draw connections between the class readings and their own writing.

RELATED WORK

All teaching techniques have cycles of popularity, and we are hardly the first to place a new emphasis on reading, writing, and rhetoric. Our new contribution in this work is the synthesis of many existing techniques into a single literate course format. We describe our experience with this format, and report on how the various techniques support and interact with each other.

A literate approach to teaching is driven by the notion that writing and thinking are inseparable: a thought cannot exist independently of its expression. This view is epitomized by Vygotsky, who has observed that language is the "apparatus of thinking" [1]. This perspective has been applied to the classroom, using the act of writing itself [2,3,4] as a vehicle for instruction.

Many have observed that technology has negatively influenced the ability to think, write, and speak clearly. For example, Sherry Turkle makes this observation of the word processor: "*the ability to quickly fill the page, to see it before you think it, can make bad writers even worse*" [5]. In a similar manner, Edward Tufte is well known for decrying the mind-numbing presentation of bulleted lists in PowerPoint: "*There's no bullet list like Stalin's bullet list!* [6]

[1] Douglas Thain, Department of Computer Science and Engineering, University of Notre Dame, dthain@cse.nd.edu
[2] Christian Poellabauer, Department of Computer Science and Engineering, University of Notre Dame, cpoellab@cse.nd.edu

1-4244-0257-3/06/$20.00 © 2006 IEEE **October 28 – 31, 2006, San Diego, CA**

Our observations of the weaknesses of new graduate students are consistent with previous observations on the differences between novice and expert writers [7]. In particular, *getting to the point* is a key skill of expert writers.

Sutinent and Tarhio emphasize "problem management"; the need for developing the ability of students to recognize and formulate both problem statements and solutions beyond the context of numbered textbook problems [8].

We note that attention to reading and writing has received less attention at FIE in recent years. In 1998, there were no less than three full sessions on writing in and across the curriculum (e.g., Barbara Olds' survey in [9]). The highly literate course project combines wisdom from many different sources in each stage of the project. Wheeler and McDonald [10] describe how writing improves collaborative activities. Reed gives a formal description of a literature review and annotated bibliography suitable as a guide for students [11]. Olds describes how draft papers can be used to improve writing quality [12]. Sharp describes the subtleties of grading of student "conferences" of a similar format to our own [13].

COMMON WEAKNESSES AMONG STUDENTS

We have had the opportunity to observe graduate students engaging in written and oral communication in many stages of their career. In the early years of the degree, students must write several papers each semester describing semester-long class projects. After joining a research project, they must regularly give short lectures describing their methods and results to other faculty and students in the department. As they gain experience, they will start to write formal research papers to be read by others in the academic community. In the final stages of education, they participate in writing persuasive research proposals sent to funding agencies. Finally, they must write a dissertation and pass several oral exams.

By the later stages of their career, most students are skilled at communication. However, many struggle in the early stages of school, particularly in classes and when writing the first few research papers. Anecdotally, we have observed the following problems:

Failure to give motivation and context. We regularly review student papers that describe interesting technical work, but fail to explain why the technical work is relevant or useful to the world at large. This is often because students choose projects by starting with a thought on the order of "*Wouldn't it be cool if...*" We continually remind students that it is not enough to do something that is *cool*: they must identify a problem, demonstrate that existing solutions are not sufficient, and only then describe a new technique and evaluate its strengths and weaknesses. This is not merely a rhetorical formality: it is necessary to guide technical decisions and later evaluation.

For example, consider the (hypothetical) students that chose a course project by asking "*Wouldn't it be cool if we parallelized data transfer in a web browser?*" Indeed, it would be cool: but who will benefit from this idea? Will this support home users watching movies, or scientific users moving around large datasets? The distinction controls the technical solution. If it will support watching a movie as it is downloaded, it must be sequentially accessed, so the parallelism must be fine-grained. If it is a large dataset that must be moved in its entirety, then the parallelism may be coarse-grained, improving throughput at the expense of response time.

Failure to get to the point. Students often have difficulty distilling a complex solution down to a simple idea. The ability to get the point is critical. When writing a paper, students must express the central idea clearly and concisely in the abstract, otherwise the paper will not get read. When attending a professional meeting, students must be able to explain their work to peers or potential employers in a matter of a few sentences, otherwise the listener will quickly tune out. When performing a literature survey, students must be able to compress long expositions into brief results, otherwise they will not be able to gain a broad view of a research area.

Consider the (hypothetical) student that wished to explore power management in laptop computers by the selective routing of packets through cross-layer introspection that identifies packets before the DMA controller transfers them to memory, but only when the machine is in a critical resource management state determined by an autonomic policy manager informed by a dialog box controlled by an interactive user. After some head scratching, the instructor said, "*So, you want the machine to start dropping packets when the battery is low?*" Students are often insulted the first time this occurs: a simple explanation can make a clever technique seem prosaic. However, we often gently remind students that it is often the simplest ideas that are the most earth shattering: consider Newton's laws of motion or Bohr's model of the atom. A simple idea can have complex consequences and yet still be easily explained.

Distraction with irrelevant details. Many students write or speak at length on details that have little meaning to anyone not intimately involved with the work. For example, they may describe the names of computers used to run experiments, the command line options necessary to run their software, or incidental technical names (LDR.EXE) rather than a descriptive name ("the loader process"). Of course, there are appropriate times to include such detail, such as when describing how to repeat a particular experiment. However, such details must be chosen carefully and employed sparingly, otherwise the work as a whole will be incomprehensible to the reader separated from the writer by any significant time or distance.

Imprecision in vocabulary. Many students are accustomed to using a variety of words -- performance, reliability, efficiency, bandwidth -- as interchangeable pleasantries describing the *goodness* of a system. We must constantly remind them that *words have meaning* and must be chosen carefully to communicate a precise idea.

A LITERATE APPROACH

To attack these weaknesses, we are developing a literate approach to core technical courses for new graduate students. Because the course format is considerably different than what

most students are accustomed to in college, we take care to establish proper expectations at the beginning of the semester. We emphasize to the students that they are pursuing a doctorate in *philosophy*, not a doctorate in *technology*, and that the work will reflect this distinction. The literate approach emphasizes the craft of scholarship alongside the technical material to be taught. The approach has two components: continuous critical evaluation of course readings, and a highly literate course project.

Critical evaluation of readings. Many graduate classes in computer science rely on a collection of recent short technical papers; the field moves rapidly and so textbooks are rarely available for advanced topics. Each class period focuses on a discussion of one or more previously assigned technical papers. Students are strongly encouraged to form study groups and discuss each paper before class. Although the primary intent of each class is to assist students with absorbing the technical material, some time at the end of each class is dedicated to exploring the rhetoric of each paper. As we carefully point out to the students, technical papers vary widely in quality of exposition; the readings serve as both positive and negative examples.

For each of the weaknesses in exposition described above, we have a series of exercises repeated throughout the semester that address the four weaknesses described above. For example, a favorite exercise is simply to ask "*Who can summarize this paper in one sentence?*" (This is often quite difficult for a 20-page technical paper.) Rarely does the first attempt by a student succeed in capturing the nuances of the paper, so several rounds of oral refinement by several students are usually necessary. Eventually, we arrive at the following sort of summary:

- *Consistency management at the granularity of a file improves the scalability of a filesystem at the cost of single-client performance.*
- *Time in a distributed system can only be explained in terms of event causality.*

Obviously, the distilled argument is only a placeholder for the complete work. Readers familiar with computer systems will recognize these as summaries of seminal papers describing AFS [14] and Lamport Clocks [15], while others would gain little insight from such a brief explanation.

Nearly every paper more than ten years old has some humorously inappropriate detail to be pointed out as an example. For example, we emphasize any mention of a powerful computer with a ten megahertz processor and an entire megabyte of memory. Students quickly get that such details become irrelevant very quickly; significant results must transcend the technical details of the day.

Some time at the end of each class is left to a discussion of the writing quality of each paper, typically by comparing with previously assigned papers. Note that it is *necessary* to leave this to the end of the class, because students often come to class with an incorrect understanding! After gaining deeper knowledge, they may change their evaluation: a paper may be easy to read and yet not effectively communicate key details. By several weeks into the semester, many students have identified a *favorite* paper against which all others must be compared.

Highly literate course project. In addition to the technical readings, students must perform a highly literate course project. The end product is a highly polished paper similar to the course readings that could plausibly be submitted to a conference or a journal. Although every project involves the creation of some software or hardware artifact, it is only the final paper that constitutes the lasting contribution. To emphasize this, we refuse to grade, collect, or even view any artifacts created by the students. This is in stark contrast to most undergraduate computer science classes, where program source code is usually the graded item, occasionally accompanied by a lab report.

The course project is developed through a series of milestones designed to encourage reflection on the final product of a research paper. Each student must research existing literature, write a formal project proposal, give a midterm progress report, write a draft paper, perform peer review, and then submit a final paper. The milestones are a deliberate attempt to provide multiple opportunities for evaluation in each mode of communication. In addition, the milestones reflect the student's graduate career in miniature - they must familiarize themselves with an area, perform solid technical work, and then spend time disseminating the results.

Annotated Bibliography. Students begin by performing a literature search and writing an annotated bibliography on a relatively broad topic such as "distributed filesystems." For many students, this is the first time they have been asked to do this activity, so we provide a fair amount of structure and guidance. For a given class, the instructor selects a handful of reliable conference proceedings and journals as canonical sources. Students must find a fixed number - say, twenty - of relevant citations and briefly summarize them. Of course, for a two-week assignment, we cannot expect students to absorb each paper in its entirety. Instead, they must quickly skim the paper to glean the idea presented, the evaluation method, and the general results. The goal of this stage is not to make students an expert in a given area, but to give them some comfort with reliable sources, and to give them a feel for the style and content of good technical papers.

Project Proposal. Having gained some breadth in a given area, students may then pick a semester project from a list generated by the instructor. This must include a description of a problem and its relevance to society, a brief survey of related work from the annotated bibliography, a description of the solution technique, a process for evaluating the technique, a rough timeline of work, and expected results. Once written, students then meet with the instructor to discuss the scope and details of the project.

Students almost always propose far more than can actually be accomplished in a semester project. This is often due to an engineering mindset: being accomplished programmers, they wish to create a large software system. However, in the academy as in the real world, large systems usually sink under the weight of bugs and implementation problems, preventing students from ever reaching an

evaluation stage. We strongly encourage students to undertake projects that involve a small amount of code and a large amount of evaluation. A common example is this: a student wishes to design a new operating system kernel from scratch that will yield spectacular networking performance. This would be an ambitious task even for a Ph.D. thesis. Such a project can easily be scaled down: we encourage such students to make a small modification to an existing operating system. One semester is barely enough time to allow for several iterations of modification, debugging, and measurement.

Progress Report. Procrastination is part of human nature, so we require a progress report to be done even before the semester is halfway complete. For this report, each team must give a ten-minute presentation of their work in front of the class, describing the overall project and the work accomplished so far. To permit everyone enough time to present, we must be ruthless in the application of a stopwatch. Students learn very quickly that giving a short talk is much harder than giving a long talk; some do not come anywhere close to completion in ten minutes. If this happens, they have an opportunity to try again at the end of the semester.

Elevator Talk. Throughout the semester, we make a habit of cornering students in the hallway or the elevator and asking the dreaded question, *"So, what are you working on?"* The student then has about thirty seconds to give an overview of the project: this is practice for *getting to the point*. Few students get this right the first time.

At the beginning of the semester, responses are generally mumbled collections of technical nouns: *distributed storage... backup... distributed failure detection...* With practice, these responses become a few complete sentences: *Backing up data to tape is too labor-prone. We are building a backup system that makes use of idle disks on workstations, which will be easier to use. The challenge is that workstations fail frequently. We deal with this by making lots of copies.*

We must confess that this practice is not universally appreciated. Some students gleefully rise to the challenge and are eager to practice their improved summaries by cornering the instructors. Other students are uncomfortable with this task. (This is similar to the concept of performance-as-exam [16].) Thus, we do not grade this activity.

Draft Paper. Several weeks before the semester ends, students write a draft paper. This must be a serious attempt at a research paper similar in form to those read in class. Students are asked to emulate the style and organization of a favorite paper from the class readings. As it is likely that not all the technical work is complete, students are permitted to leave out experimental results. However, they must still describe the experimental work, provide graphs with axes, labels, and captions, and explain the significance of possible results, whether positive or negative.

Peer Review. Immediately after handing in the draft paper, students are asked to perform peer reviews of each others drafts. In this stage, papers are exchanged across two simultaneous classes. This has two effects: first, the material will be fresh to the reader, who has already heard his/her classmates explain their work one or more times. Second, it forces the writer to provide enough introductory material so that readers in the same field but not the same area of expertise can understand the paper. A form shown in Figure 1 is given to guide the reviewer. This is made available to students writing the draft papers so that the expectations are clear. The form is similar to those used in peer review of scientific work, leaving out the accept/reject decision. Students must write about one page of text for each paper reviewed. Once written, the reviews are collected by the instructor, who grades the *reviews* for quality, collects them together and writes a very brief (i.e. minimal marking [17]) summary pointing out what must be improved in the final paper. All are returned to the student for consideration.

SUMMARY
In one solid paragraph, summarize the contributions of this paper. This serves to demonstrate that the reviewer actually read and understood the paper. It also helps the author to see what points attract the reader's attention.

BACKGROUND AND RELATED WORK
- Does the author lay out a clear and convincing rationale for the work?
  - If not, how could be rationale be improved?
- Is the introduction accessible to a computer scientist in a different field?
  - If not, what needs to be elaborated to assist the non-expert?
- Have appropriate works been cited and described correctly?
  - If not, what should be added or removed?

TECHNICAL CONTENT
- Do the proofs/algorithms/structures accomplish what the authors claim?
  - If not, exactly what is wrong?
- Have the authors considered all of the consequences of their work?
  - If not, what needs to be considered?
- What evidence is given that the system is built and works?
  - If none, what could be given?

EVALUATION
- What measurements have the authors made?
- What conclusions have they drawn?
- Does the evaluation back up claims made earlier in the paper?
  - If not, what must be done to rectify the evaluation?

COMMUNICATION
- Is the organization of the paper suitable?
  - If not, suggest how it could be re-arranged.
- Are technical concepts explained clearly and accurately?
  - If not, suggest how they may be improved.
- Is the paper carefully proofread and spell-checked?
  - If not, point out a few places that need attention.

FIGURE 1. PEER REVIEWING FORM

Final Paper. Using the feedback from the review process, students revise the final paper. We emphasize to the students that revision does not mean fixing a few typos. (Reviewers are also discouraged from focusing on typographical errors.) Rather, the process of revising a paper should be considered major surgery: sections should be reorganized, technical terminology reconsidered, motivating arguments changed, even the title of the paper should be refined. Students submit the final paper along with the draft in order to demonstrate that serious changes have been made.

Final Lecture. As the last activity of the semester, students give a final twenty minute lecture describing the project. Although students receive a grade for the final lecture, this session consistently has a light-hearted mood. The students finally have results to show, are comfortable with well-practiced material, and happy to be at the semester's end.

About how much time did you spend writing your draft paper?
Average: 10.9 ± 5.2 hours
About how much time did you spend writing your final paper?
Average: 10.6 ± 7.6 hours
About how much time did you spend reading each paper?
Average: 1.9 ± 1.0 hours
About how much time did you spend writing each review?
Average: 1.3 ± 0.9 hours
On average, how **difficult** was it to read your peer's papers?
1- Trivial - One quick read straight through.
2- Easy - A few details required a quick review.
3- Moderate - Several important details required careful study.
4- Difficult - Several readings were necessary to understand the main point.
5- Incomprehensible - Most of the paper was inaccessible.
Average: 2.9 ± 0.5
How **thorough** were the reviews of your paper?
1 - Almost every single paragraph received comments.
2 - Almost every section of the paper received comments.
3 - The most important points received comments.
4 - Only a few incidental details received comments.
5 - I'm not sure if the reviewers read the paper at all.
Average: 2.8 ± 0.5
How **accurate** were the reviews of your paper?
1 - The reviewers understood every detail perfectly.
2 - The reviewers didn't understand minor details.
3 - The reviewers didn't understand important details.
4 - The reviewers didn't understand the main point.
5 - The reviewers didn't understand anything.
Average: 2 ± 0.6
(N=34)

FIGURE. 2. QUANTITATIVE SURVEY RESULTS

On the utility of performing a literature search and refining a project proposal before beginning to work:
- [I learned that] it takes many hours/days/weeks to refine a topic to something neat, useful, and innovative.
- I learned to quickly identify the usefulness of a paper...
- The proposal must be as generic as possible, so it won't restrict your further course.

On the relevance of reading and discussing research papers to writing your own paper:
- I tried to make my writing less dense than some because some [of the papers that we read] were so dense that they inhibit comprehension.
- It showed me how to attract readers.
- More discussion about what makes a paper good would be excellent.

On the value of writing a draft paper before the final paper:
- It made our group actually start on the project and get some work accomplished.
- It cut down on the time needed for a final version.
- Every time I read my paper, I find more to polish it.

On the process of performing peer reviews:
- Thinking critically about another's work made it much easier to reconsider the effectiveness with which we were presenting our results.
- It makes you realize common sources of error or concern, which often are in your own paper, but you don't realize because you have read it too many times already.
- I did pick up a few points that were helpful revisions, but the reviewers generally have no hard-hitting comments. I would have preferred tougher reviews.

On the literate approach overall:
- I became more critical of my own work.
- Helped me to formalize what a "good" paper was.
- The process is very instructive but is also an incredible amount of work.
- Consumes a lot of time.

FIGURE 3. SELECTED STUDENT REFLECTIONS

EVALUATION

This course format was conducted in fall 2004 in two small advanced elective graduate courses: Advanced Computer Architecture and Distributed Systems, totaling 11 students. In fall 2005, the same format was used again in two courses for beginning graduate students: Operating Systems and Real-Time Systems, totaling 36 students.

The primary goal of the literate approach is to better prepare students for professional scholarship. To this end, over the four courses, a total of five term projects were accepted as peer-reviewed papers at scientific conferences – two in 2004 and three in 2005. We hope to see this number increase in future years.

In fall 2005, we performed an end-of-semester evaluation via required but anonymous student surveys at the end of the semester across both classes. This evaluation had two components: The first component was a set of quantitative questions to measure the effort spent on various components of the course, and the relative value of reading and writing peer reviews. Our major concern is designing the course is that the peer reviews would be of poor quality or misdirected, thus many questions focus on the effort and quality of reviewing. A summary of the responses to the first component are shown in Figure 2.

The second component of the survey was a set of open-ended questions designed to elicit reflections on the strengths and weaknesses of the literate approach. The purpose of this exercise was to gauge whether students had absorbed the primary lesson of the importance of rhetoric as a professional skill. We subjectively selected a set of responses that were clearly written and expressed a range of opinion and reproduced them in Figure 3.

The instructors met several times during each semester and again at the end to record and exchange observations on student performance in response to the literate approach. A discussion of these observations and several mid-course corrections are given in the following section.

DISCUSSION

The quantitative survey questions allow us to calibrate the relative utility of the course components. The instructors were pleasantly surprised that the students indicate that the average times spent on writing the draft paper and the final paper are about the same. (Note that both draft and final papers were graded, albeit with different weights.) Further, the majority of students perceived the difficulty of reading their peers' papers as moderate, which can be explained with a combination of too much technical detail (by the author) and inexperience in writing and reviewing papers (by both the author and the reviewer). (Remember that peer reviews are performed across different courses!) Finally, the average response to the thoroughness and the accuracy of the reviews were "*The most important points received comments*" and "*The reviewers didn't understand minor details.*" Both reflect that students were able to provide comments on relevant aspects of the paper and grasp the key ideas, novelties, and importance of the project, even though it was the first time for most students to participate in a review process.

From the student comments, it appears that *writing* the reviews may have had more value than *reading* the reviews! While absorbing papers written by others, the students were able to identify weaknesses in their own papers. However, as one comment states, few reviewers made any hard-hitting comments. Although we would certainly like to encourage students be more critical, this is an area in which we must tread carefully: unchecked criticism can be damaging both to the reader as well as the writer. To present a model of appropriate reviewing, it would be helpful to share actual reviews for real research papers. Unfortunately, the confidential nature of reviews makes this difficult.

The regular instructor meetings yielded several observations. Anecdotally, the instructors observed that the students' skill at explaining their chosen projects both in and out of the classroom had improved dramatically over the course of the semester. However, more systematic assessment is needed across the curriculum to see whether these gains will also be reflected as the students proceed through graduate school. We also found that students were much more comfortable proceeding when given a model result for an assignment; thus the students in the second semester were able to view the projects and results produced by students in the earlier semesters.

Several challenges were noted across the courses. First, it is difficult to strike the proper balance between emphasizing the final product and encouraging continuous development. In the first year that this format was offered, the final paper was a significant fraction of the course grade, while the draft paper was only a few token points. As might be expected, this resulted in generally poor drafts, but a serious last-minute effort on the final paper. In the second year, we gave equal weight to the draft and final papers, but this resulted in several final papers that were not significantly revised from the draft. One potential solution is to grade the draft paper using the criteria for the final paper, so that most students receive a C or D for the draft, and then have some sense of urgency regarding the final paper. Of course, this means that many students receive a poor grade for what is actually a respectable start on the paper. If the final paper grade is used to replace the draft grade, then there is little incentive to produce a good draft. There may not be an ideal solution to this problem.

Second, there is always some tension between peer reviewing and the instructor's grading criteria. In the first year, each instructor provided a detailed review alongside the peer reviews. This was a mistake - students quite reasonably followed the instructor's advice and ignored the peer's advice. In the second year, we moved to the model of providing only peer reviews augmented with the "referee's" summary. The summary was simply a short list of priorities, allowing the instructor to point out repeated themes among the reviewers and negate the occasional bad advice. This also corresponds better to what happens in practice with academic journals. Despite having provided a reviewing form and a fair amount of guidance, a few students did not do a good job of reviewing and simply pointed out a few typos. Fortunately, most students received one or two quality reviews. In two cases where students did not receive any quality reviews, the instructors provided more detailed referee comments.

Finally, the overall workload of this approach is very heavy for first-year students. There are several ways that we may reduce the workload without significantly altering the course format. Several lectures allocated to course readings could be removed, leaving more time for discussion of the course project and paper writing process. The project proposal and the progress report might be combined into one assignment. The peer reviewing could be reduced to fewer papers read by each student, but this increases the risk of a student receiving no useful reviews. Better policing of project choices by the instructors may result in simpler projects with a lighter workload.

CONCLUSION

The skills of reading, writing, and rhetoric were once considered the core of a classical education, which has all but disappeared from the American educational landscape in the last century. The literate approach is an attempt to return some elements of a classical education to the modern classroom.

REFERENCES

[1] L. Vygotsky. *Thought and Language*. MIT Press, 1986.

[2] J. Enig, *College Composition and Communication*, 28:122-128, 1977.

[3] J. Auerbach, C. Bouregois, T. Collins. Do students benefit? writing-to-learn in a digital design laboratory course. In ASEE/IEEE FIE 2004.

[4] D. Larson, S. Gruber, D. Scott, and M. Neville. A holistic assessment of writing in design. In ASEE/IEEE FIE 1998.

[5] S. Turkle. Howe computers change the way we think. *Chronicle of Higher Education*, 50:B26, January 2004.

[6] E. Tufte. *The Cognitive Style of Powerpoint: Pitching out Corrupts Within.* Graphics Press, Cheshire, CT, 2006.

[7] O. Clua and M. Feldgen. Difference among experts, novices, and trainees in writing a report. In ASEE/IEEE FIE 2004.

[8] E. Sutinen and J. Tarhio. Teaching to identify problems in a creative way. In ASEE/IEEE FIE 2001.

[9] B. Olds. Technical writing across the curriculum: process, problems, and progress. In ASEE/IEEE FIE 1998.

[10] E. Wheeler and R. McDonald. Using writing to enhance collaborative learning in engineering courses. In ASEE/IEEE FIE 1998.

[11] L. Reed. Perfoerming a literative review. In ASEE/IEEE FIE 1998.

[12] B. Olds. Using draft revisions to improve writing and thinking in engineering courses. In ASEE/IEEE FIE 1994.

[13] J. Sharp. Grading technical papers during student conferences. In ASEE/IEEE FIE 1994.

[14] J. Howard, M. Kazar, S. Menees, D. Nichols, M. Satyanarayanan, R. Sidebotham, and M. West. Scale and performance in a distributed file system. ACM Trans. on Comp. Sys., 6(1):51-81, February 1988.

[15] L. Lamport. Time, clocks, and the ordering of events in a distributed system. Communications of the ACM, 21(7):558-564, July 1978.

[16] B. London and L. Deyo. The ballet model in engineering classes – what works, what doesn't, and what's new. In ASEE/IEEE FIE 2005.

[17] R. Haswell. Minimal marking. *College English*, 45(6):166-170, 1983.