On Teaching Computer Ethics within a Computer Science Department

Michael J. Quinn
School of Electrical Engineering and Computer Science, Oregon State University, USA

Keywords: ABET, accreditation, ethics, computer science, teaching

ABSTRACT: *The author has surveyed a quarter of the accredited undergraduate computer science programs in the United States. More than half of these programs offer a "social and ethical implications of computing" course taught by a computer science faculty member, and there appears to be a trend toward teaching ethics classes within computer science departments. Although the decision to create an "in house" computer ethics course may sometimes be a pragmatic response to pressure from the accreditation agency, this paper argues that teaching ethics within a computer science department can provide students and faculty members with numerous benefits. The paper lists topics that can be covered in a computer ethics course and offers some practical suggestions for making the course successful.*

1. Introduction

About 200 colleges and universities in the United States have baccalaureate computer science programs accredited by ABET's Computing Accreditation Commission. In order to receive accreditation, a program must meet various intents. Typically, an institution demonstrates that it meets an intent by showing how it satisfies the standards associated with the intent.

Standard IV-17 states, "There must be sufficient coverage of social and ethical implications of computing to give students an understanding of a broad range of issues in this area."[1] The author has polled 50 departments with accredited computer science programs to learn how they meet this standard. The detailed results of the poll appear in the Appendix (see pp. 341-342).

An earlier version of this paper was presented at the 2005 conference, *Ethics and Social Responsibility in Engineering and Technology, Linking Workplace Ethics and Education*, co-hosted by Gonzaga University and Loyola Marymount University, Los Angeles, California, 9-10 June 2005.

Address for correspondence: Michael J. Quinn, School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331, USA; email: quinn@eecs.oregonstate.edu.

Paper received, 28 October 2005: revised, 12 January 2006: accepted, 2 February 2006.

Thirty percent of the surveyed departments meet the standard by incorporating discussions of social and ethical issues of computing within other computer science courses. The most popular places in the curriculum for including discussions of social and ethical issues are: software engineering, the senior project, a computer science orientation class (CS 0), or an introductory programming course (CS 1 or CS 2).

Fifteen percent of the surveyed departments meet the standard by requiring computer science majors to take an ethics course taught by another department, typically philosophy.

Fifty-five percent of the surveyed departments require computer science majors to take a social and ethical issues of computing course taught within the computer science department. In most cases the course is worth three semester hours of credit and is taken by juniors or seniors.

In the process of conducting phone interviews, the author learned that several departments have created their own computer ethics course for a pragmatic reason: during a previous accreditation visit they had a difficult time demonstrating they had met Standard IV-17. According to the department chairs the author interviewed, program evaluators want to see graded student work related to the standard. When ethics is taught across the curriculum, it is more difficult to guarantee that instructors will actually create homework assignments or test questions related to the social and ethical aspects of computing. Even if instructors do create these assignments and exam questions, it can still be difficult for the department to gather the evidence. Also, department chairs reported that ABET program evaluators would like students to be exposed to moral issues related to information technology. A generic ethics course taught within a philosophy department is unlikely to spend enough time on information-technology-specific case studies. For these reasons, creating a separate computer ethics class makes it much easier for a department to demonstrate that its undergraduate program satisfies Standard IV-17.

The accreditation agency is right to expect departments to provide evidence they are giving students significant exposure to the social and ethical implications of computing, and the trend toward computer science departments offering their own ethics class is a positive one. Teaching a “computers and society” class can provide many benefits, which will be elaborated in the next section.

2. Should Computer Science Professors Teach Computer Ethics?

Service courses are a familiar feature on the college landscape. Many departments, such as mathematics, generate far more student credit hours teaching classes to non-majors than to majors. If a computer science department “farms out” its math classes to the mathematics faculty, why shouldn’t it let the philosophy faculty handle the ethics class?

There is a fundamental difference between the two situations. Even if computer science students take math classes taught by the mathematics department, math is still used in many computer science classes. The two experiences complement each other. Students learn mathematical theory in their math classes, and they learn how to apply the theory in their computer science classes. As a whole, computer science professors are comfortable "doing mathematics" in their classes, which is not surprising, considering a significant proportion of computer science faculty members aged 50 and older majored in mathematics as undergraduates. In contrast, many computer science faculty members are reluctant to raise moral issues in the context of the computer science classes they are teaching. That being the case, there is the very real possibility that if ethics is taught by faculty members in philosophy, computer science students may never have the experience of seeing a computer science professor "doing ethics." When computer science faculty members teach ethics, they serve as role models who demonstrate that contemplating the ethical dimensions of everyday decisions is something that everyone can and should do — not just those with a Ph.D. in philosophy.

Another advantage to teaching ethics "in house" is that more time can be spent discussing issues that are confronted by professionals in the computing field. Participating in the analysis of such issues can help students become more alert to ethical concerns and give them the tools they need to evaluate workplace situations.

Still, there are obstacles to creating an ethics class inside a computer science department. One obstacle is the objection that computer science faculty members should not be in the business of teaching values to students. Rebutting this objection, James Moor suggests that computer science professors should not be jittery about teaching values, because they teach values all the time.[2] For example, an important part of teaching programming is helping students understand what distinguishes a "good" program from a "bad" program. A well-documented, well-structured, efficient, correct program is superior to an inefficient, undocumented piece of spaghetti code that does not produce correct results.

Computer science professors may reply that they are not nervous about teaching values, just *moral* values. However, computer scientists do enter into philosophical debates. Consider Edsger Dijkstra's famous letter "Go to statement considered harmful," that appeared in the March 1968 issue of *Communications of the ACM*. His letter began with the inflammatory statement, "For a number of years I have been familiar with the observation that the quality of programmers is a decreasing function of the density of '`go to`' statements in the programs they produce."[3] I suspect most computer scientists, at one time or another, have engaged in "religious debates" about the superiority of one programming language or construct over another. When you cannot demonstrate the superiority of one alternative over another with a mathematical proof or a statistically significant result from a controlled study, you must rely on other means, such as logic, the evidence you do have at your disposal, and commonly held values. These are the same resources that must be used when debating philosophical issues.

A more substantial obstacle to teaching ethics inside a computer science department is reflected in the statement, "I don't know anything about teaching a computer ethics class." There are two facets to this reservation. One has to do with the content of a computer ethics class; the other has to do with its form.

With respect to the content of a computer ethics class, it all boils down to how much time a faculty member spends studying the material and preparing for the class. The time commitment can be reduced substantially if the faculty member's higher education included a few philosophy classes. Even more valuable is the presence of helpful colleagues among the philosophy faculty who are willing to clarify points of confusion. Virtually every college that has computer science faculty also has faculty in philosophy.

The other facet of the reservation has to do with the form of a computer ethics class. Compared to a typical computer science class, a computer ethics class is much more discussion-oriented. The homework assignments and examinations are quite different, too. Fortunately, faculty members new to this kind of class can get useful advice from their colleagues in the liberal arts, as well as computer ethics instructors at other colleges and universities.

In both cases the real issue is whether expertise relevant to teaching computer ethics is valued and rewarded within a computer science department. Faculty members will be much more motivated to improve their understanding of ethical theories and their ability to manage discussion-oriented classes if professional ethics is considered a legitimate sub-discipline when salary, promotion, and tenure decisions are being made.

Faculty members who diligently prepare for and teach a computer ethics course will become more comfortable discussing ethical issues. Since it is likely that faculty members assigned to ethics courses also teach other computer science courses, there is the opportunity for them to raise ethical concerns in these courses, when appropriate. This may be the most important and welcome consequence of teaching ethics inside a computer science department. Again, the integration of professional ethics into the rest of the curriculum is more likely to occur if scholarly work in this area is valued by the department. As Bynum puts it, "Computer ethics could not be considered 'something extra,' an after thought to be 'added on' to the curriculum."[4]

3. Helpful Techniques for Teaching Computer Ethics

A course in computers, ethics, and society has been part of the computer science curriculum at Oregon State University for more than a decade. During this time, I have had the opportunity to teach this class nine times. I have also had the opportunity to learn from the experiences of other computer ethics instructors, both at OSU and at other colleges and universities. In this section I present some "lessons learned" that may be valuable for those teaching a computer ethics course.

An important consideration is the selection of topics to be covered in the course. A joint task force of the IEEE Computer Society and the Association for Computing Machinery released a model computer science curriculum called Computing Curricula 2001.[5] In the model curriculum, every undergraduate computer science student would receive 40 hours of instruction regarding social and professional issues related to computing. The model syllabus for the CS 280T, Social and Professional Issues, contains these major topics: history of computing, social context of computing, methods and tools of analysis, professional and ethical responsibilities, risks and liabilities of computer-based systems, intellectual property, privacy and civil liberties, computer crime, economic issues in computing, and philosophical frameworks.

Once the topics have been selected, the next question is how to present the material. Studies have revealed the effectiveness of instructor-guided discussions of cases or particular situations that allow students to determine the best course of action.[6] Fruitful discussions provide an opportunity for personal growth by exposing students to points of view much different from their own. The job of the instructor is to raise questions, give students the opportunity to formulate answers, keep a few students from dominating the discussion, and provide feedback that enables the students to improve their ability to present cogent arguments and defend their conclusions. Without guidance, discussions can deteriorate into a series of "I think…" statements. By asking students to justify their positions, the instructor can reinforce the importance of bringing facts to bear and using ethical theories correctly. Under the instructor's guidance, the discussion can progress to the point where each student can determine the best course of action. Martin and Holz put it this way: "Our belief is that ethics cannot be taught; rather what can be taught is a framework for evaluating ethical dilemmas and making decisions. In accepting the premise that technology is value-laden, we stress the need to teach a methodology of explicit ethical analysis in all decision-making related to technology."[7]

A blackboard or whiteboard should be used to keep track of the major points raised and the facts or values supporting these points. Writing key points on the blackboard demonstrates to the class that every student is being listened to, even the less assertive students. It helps students determine the strength of each side to the argument. It can make it easier for students to identify contradictory points, and it reduces repetition in discussions.

The most popular activity among the students I've taught has been role-playing exercises. For example, when discussing the Therac-25 radiation therapy machine accidents in Tyler, Texas, there are six roles. Three roles are associated with "defendants:" the technician operating the machine, the hospital administration, and the programmer who wrote the code for the Therac-25.[8] There are also three roles for "moral prosecutors:" one for each of the defendants. The role of a prosecutor is to explain why their particular party is morally responsible for the deaths at the hospital. The role of the defendant is to explain why he or she is not morally responsible. I divide the class into small groups, each representing one of the roles. The groups huddle for 15-20 minutes to develop their case. These group discussions are a good way to have many productive conversations going on simultaneously. By presenting their case to the rest of the class, students improve their presentation skills and their ability to think on their feet. Sometimes I ask some of the students in the class to act as a "jury" that weighs the arguments and allocates moral responsibility among the various roles.

In contrast to role-playing exercises, debates are much less popular with students. Students are not used to rebutting each other's arguments. If each side delivers its arguments without meaningful rebuttals from the other side, it is difficult to determine who had the better argument. Debates work best when the class is more-or-less evenly divided on the issue. Otherwise, there may not be enough interesting arguments on one side of the question. Many students who feel strongly about an issue find it difficult to develop a logical case supporting their position, because they can't see beyond their emotions. One way to address this problem is to ask students who support a position to develop the case *against* the position, and vice versa. When students are not emotionally invested in the argument, it can be easier for them to focus on facts and logic.[9]

Writing assignments are critically important. Weak arguments are much easier to spot when they are written down, instead of being delivered orally. In a typical term-paper assignment, I ask students to present an ethical issue related to the contemporary or planned use of information technology, analyze the issue from the point of view of a consequentialist ethical theory, analyze it again from the point of view of a nonconsequentialist theory, and present their conclusions. When there are good arguments on both sides of an issue, many students find it hard to choose which side is stronger. I insist that students choose a side and defend their choice. A standardized scoring rubric makes grading essays easier.[10]

Classroom discussions are meatier if students actually complete the reading assignments. Short weekly quizzes encourage students to keep up with the reading. On the midterm and final examinations, multiple-choice and fill-in-the-blanks questions test the students' factual knowledge. Essay questions test their ability to construct reasonable arguments.

A course with all of these features can engage the majority of the students in the class and provide them with numerous opportunities to think more deeply about the ethical dimensions of everyday professional situations.

4. Summary

A random survey of accredited undergraduate programs in computer science has revealed that more than half of the departments teach their own "computers, ethics, and society" class. However, the survey has also provided evidence that, for at least some departments, the primary motivation for offering their own computer ethics course is to make the accreditation process go more smoothly. It would be better if departments viewed ethics as important as other core topics in the computer science curriculum.

Making ethics a priority would have a variety of beneficial consequences. Faculty members assigned to teach the ethics course would be more highly motivated to improve their understanding of ethical theories and their ability to evaluate ethical dilemmas. They would see the value in achieving a level of scholarship that would enable them to publish papers in this area. They would find new opportunities to bring up ethical issues in their other computer science classes, helping students appreciate the ethical dimensions of everyday professional situations.

Appendix: Results of Survey of 50 Accredited Computer Science Programs

In April and May of 2005, the author polled 50 departments with ABET-accredited baccalaureate programs in computer science. (For a complete list of ABET-accredited computer science programs, visit the ABET Web site: www.abet.org/accreditcac.asp.) The purpose of the poll was to answer these questions:

- What is the percentage of departments that require computer science majors to take an ethics class?
- For those departments that require an ethics class, how many of them teach the ethics class within computer science?
- For those departments that require an ethics class, what is the number of credit hours, and what is the class standing of the typical student?
- For those departments that do not require a stand-alone ethics class, in what computer science classes is ethics discussed?

The results of the poll appear in the following table. To make the results consistent, all credit hours have been converted to semester credit hours.

	Specific Ethics Class?		
College/University	**CS**	**Non-CS**	**Notes**
Arizona State U.	1 credit		Junior level
Boise State U.	1 credit		Sophomore level
Bucknell U.	2 credits		Sophomore level
Cal. State – Dominguez Hills	3 credits		
Cal. State – San Bernardino	1.33 cr.		Senior level
Calvin College	3 credits		Senior level
College of Charleston			Woven into software engineering and practicum
East Tennessee State U.	3 credits		Junior/senior level + woven into two software engineering classes
Eastern Washington State U.	3 credits		Senior level
Florida Atlantic U.	1 credit		Senior level
George Mason U.	4 credits		Freshman (1) + Junior (3)
Georgia Tech	3 credits		Senior level
Grambling State U.		3 credits	Sophomore level
Indiana U. – Purdue U. Fort Wayne			Woven into senior capstone design
Kennesaw State U.			Woven into every course
Loyola College in Maryland		3 credits	Senior level
Metropolitan State College of Denver		3 credits	Junior level
Millersville U. of Pennyslvania			Woven into CS 1, CS 2, and junior-level courses
Montana State U.	2 credits		Sophomore level

New Jersey Inst. of Technology	3 credits		Junior level
Nicholls State U.			Woven into CS 1, software engineering, and 1-2 others
Northeastern U.		4 credits	Junior/senior level
Radford U.			Woven into CS 1, senior seminar, and 2 others
North Dakota State U.	3 credits		Senior level
Oregon State U.	2 credits		Junior level
Salem State College			Woven into CS 1
Southern Illinois U., Edwardsville			Woven into CS 0, HCI, senior project
Southwest Missouri State U.			Woven into CS 0 and senior seminar
Texas Christian U.			Woven into CS 2, database
U. Alabama, Huntsville	1 credit		Teaching alternates between CS and Philosophy
U. California, Santa Barbara		3 credits	Freshman level
U. Central Florida		3 credits	Junior level
U. Colorado, Denver	3 credits		Senior level
U. Idaho	1 credit		Senior level
U. Louisiana, Lafayette	2 credits		Junior level
U. Mississippi	1 credit		Junior level
U. Nebraska, Omaha	3 credits		Junior level
U. Nevada, Reno	3 credits		Junior level
U. New Hampshire		4 credits	Junior level + junior-level CS public speaking course
U. New Orleans	1 credit		Junior level
U. Oklahoma			Woven into CS 2, data structures, GUI, software engineering
U. Pacific	3 credits		Senior level
U. South Alabama			Woven into CS 0 and senior design sequences
U. South Florida	3 credits		Senior level
U. Tennessee, Chattanooga	3 credits		Junior level
U. West Georgia			Woven into senior capstone and other courses
U. Wyoming	1 credit		Junior level
Utah State U.			Woven into CS 1, CS 2, software engineering, seminar
Virginia Commonwealth			Woven into CS 1, program design, seminar
Winthrop U.	3 credits		Junior level

REFERENCES

1. Computing Accreditation Commission. (2004). *Criteria for Accrediting Computing Programs*, ABET, Inc.
2. Moor, J.H. (1998). Reason, relativity, and responsibility in computer ethics. *Computers and Society* **28**, 1: 14-21.
3. Dijkstra, E.W. (1968). Letter to the editor: Go to statement considered harmful. *Communications of the ACM* **11**, 3: 147-148.
4. Bynum, T.W. Computer ethics in the computer science curriculum. *The Research Center on Computing & Society*. www.southernct.edu/organizations/rccs.
5. Joint Task Force on Computing Curricula. (2002). *Computing Curricula 2001, Computer Science.* IEEE Press.
6. Lisman, C. D. (1996). *The Curricular Integration of Ethics: Theory and Practice.* Praeger Publishers.
7. Martin, C.D., and Holz, H.J. Non-apologetic computer ethics education: a strategy for integrating social impact and ethics into the computer science curriculum. *The Research Center on Computing & Society*. www.southernct.edu/organizations/rccs.
8. Baase, S. (2003). *A Gift of Fire: Social, Legal, and Ethical Issues for Computers and the Internet.* Second Edition. Prentice-Hall, Upper Saddle River, NJ.
9. Grodzinsky, F., Gehringer, E., King, L., and Tavani, H. (2004). Responding to the challenges of teaching computer ethics. Panel session. Thirty-fifth SIGCSE Technical Symposium on Computer Science Education: SIGCSE 2004, Norfolk Virginia, March 3-7.
10. Moskal, B., Miller, K., and King, L. A. S. Grading essays in computer ethics: rubrics considered helpful. In *Proceedings of the Thirty-fifth SIGCSE Technical Symposium on Computer Science Education: SIGCSE 2004*, Association for Computing Machinery, pp. 101-105.