9.3 Software Engineering Code of Ethics

The Software Engineering Code of Ethics and Professional Practice is a practical framework for moral decision-making related to problems that software engineers may encounter.

The Software Engineering Code of Ethics and Professional Practice, reproduced in its entirety below, is Copyright © 2011 by ACM Publications. Reprinted with permission.

9.3.1 Preamble

Computers have a central and growing role in commerce, industry, government, medicine, education, entertainment and society at large. Software engineers are those who contribute by direct participation or by teaching, to the analysis, specification, design, development, certification, maintenance and testing of software systems. Because of their roles in developing software systems, software engineers have significant opportunities to do good or cause harm, to enable others to do good or cause harm, or to influence others to do good or cause harm. To ensure, as much as possible, that their efforts will be used for good, software engineers must commit themselves to making software engineering a beneficial and respected profession. In accordance with that commitment, software engineers shall adhere to the following Code of Ethics and Professional Practice.

The Code contains eight Principles related to the behavior of and decisions made by professional software engineers, including practitioners, educators, managers, supervisors and policymakers, as well as trainees and students of the profession. The Principles identify the ethically responsible relationships in which individuals, groups, and organizations participate and the primary obligations within these relationships. The Clauses of each Principle are illustrations of some of the obligations included in these relationships. These obligations are founded in the software engineer's humanity, in special care owed to people affected by the work of software engineers, and the unique elements of the practice of software engineering. The Code prescribes these as obligations of anyone claiming to be or aspiring to be a software engineer.

It is not intended that the individual parts of the Code be used in isolation to justify errors of omission or commission. The list of Principles and Clauses is not exhaustive. The Clauses should not be read as separating the acceptable from the unacceptable in professional conduct in all practical situations. The Code is not a simple ethical algorithm that generates ethical decisions. In some situations standards may be in tension with each other or with standards from other sources. These situations require the software engineer to use ethical judgment to act in a manner which is most consistent with the spirit of the Code of Ethics and Professional Practice, given the circumstances.

Ethical tensions can best be addressed by thoughtful consideration of fundamental principles, rather than blind reliance on detailed regulations. These Principles should influence software engineers to consider broadly who is affected by their work; to examine if they and their colleagues are treating other human beings with due respect; to consider how the public, if reasonably well informed, would view their decisions; to analyze how

the least empowered will be affected by their decisions; and to consider whether their acts would be judged worthy of the ideal professional working as a software engineer. In all these judgments concern for the health, safety and welfare of the public is primary; that is, the "Public Interest" is central to this Code.

The dynamic and demanding context of software engineering requires a code that is adaptable and relevant to new situations as they occur. However, even in this generality, the Code provides support for software engineers and managers of software engineers who need to take positive action in a specific case by documenting the ethical stance of the profession. The Code provides an ethical foundation to which individuals within teams and the team as a whole can appeal. The Code helps to define those actions that are ethically improper to request of a software engineer or teams of software engineers.

The Code is not simply for adjudicating the nature of questionable acts; it also has an important educational function. As this Code expresses the consensus of the profession on ethical issues, it is a means to educate both the public and aspiring professionals about the ethical obligations of all software engineers.

9.3.2 Principles

PRINCIPLE 1: PUBLIC

Software engineers shall act consistently with the public interest. In particular, software engineers shall, as appropriate:

1.01 Accept full responsibility for their own work.

1.02 Moderate the interests of the software engineer, the employer, the client and the users with the public good.

1.03 Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good.

1.04 Disclose to appropriate persons or authorities any actual or potential danger to the user, the public, or the environment, that they reasonably believe to be associated with software or related documents.

1.05 Cooperate in efforts to address matters of grave public concern caused by software, its installation, maintenance, support or documentation.

1.06 Be fair and avoid deception in all statements, particularly public ones, concerning software or related documents, methods and tools.

1.07 Consider issues of physical disabilities, allocation of resources, economic disadvantage and other factors that can diminish access to the benefits of software.

1.08 Be encouraged to volunteer professional skills to good causes and contribute to public education concerning the discipline.

FIGURE 9.2 Software engineers shall approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy, or harm the environment. The ultimate effect of the work should be to the public good (Clause 1.03).

PRINCIPLE 2: CLIENT AND EMPLOYER

Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest. In particular, software engineers shall, as appropriate:

2.01 Provide service in their areas of competence, being honest and forthright about any limitations of their experience and education.

2.02 Not knowingly use software that is obtained or retained either illegally or unethically.

2.03 Use the property of a client or employer only in ways properly authorized, and with the client's or employer's knowledge and consent.

2.04 Ensure that any document upon which they rely has been approved, when required, by someone authorized to approve it.

2.05 Keep private any confidential information gained in their professional work, where such confidentiality is consistent with the public interest and consistent with the law.

2.06 Identify, document, collect evidence and report to the client or the employer promptly if, in their opinion, a project is likely to fail, to prove too expensive, to violate intellectual property law, or otherwise to be problematic.

2.07 Identify, document, and report significant issues of social concern, of which they are aware, in software or related documents, to the employer or the client.

2.08 Accept no outside work detrimental to the work they perform for their primary employer.

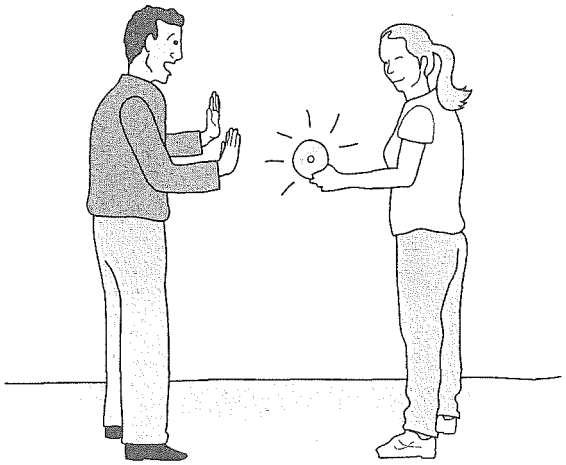

FIGURE 9.3 Software engineers shall not knowingly use software that is obtained or retained either illegally or unethically (Clause 2.02).

2.09 Promote no interest adverse to their employer or client, unless a higher ethical concern is being compromised; in that case, inform the employer or another appropriate authority of the ethical concern.

PRINCIPLE 3: PRODUCT

Software engineers shall ensure that their products and related modifications meet the highest professional standards possible. In particular, software engineers shall, as appropriate:

3.01 Strive for high quality, acceptable cost and a reasonable schedule, ensuring significant tradeoffs are clear to and accepted by the employer and the client, and are available for consideration by the user and the public.

3.02 Ensure proper and achievable goals and objectives for any project on which they work or propose.

3.03 Identify, define and address ethical, economic, cultural, legal and environmental issues related to work projects.

3.04 Ensure that they are qualified for any project on which they work or propose to work by an appropriate combination of education and training, and experience.

3.05 Ensure an appropriate method is used for any project on which they work or propose to work.

3.06 Work to follow professional standards, when available, that are most appropriate for the task at hand, departing from these only when ethically or technically justified.

3.07 Strive to fully understand the specifications for software on which they work.

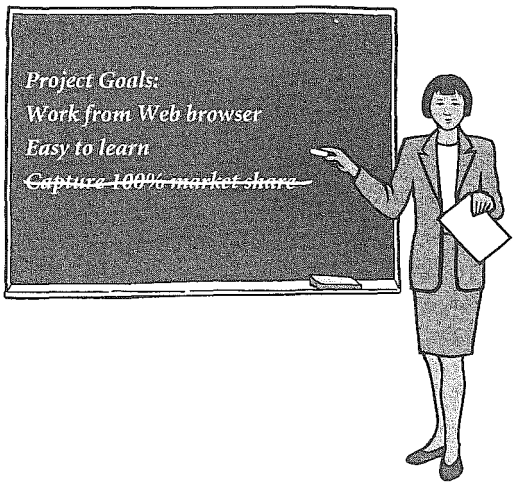

FIGURE 9.4 Software engineers shall ensure proper and achievable goals and objectives for any project on which they work or propose (Clause 3.02).

3.08 Ensure that specifications for software on which they work have been well documented, satisfy the users' requirements and have the appropriate approvals.

3.09 Ensure realistic quantitative estimates of cost, scheduling, personnel, quality and outcomes on any project on which they work or propose to work and provide an uncertainty assessment of these estimates.

3.10 Ensure adequate testing, debugging, and review of software and related documents on which they work.

3.11 Ensure adequate documentation, including significant problems discovered and solutions adopted, for any project on which they work.

3.12 Work to develop software and related documents that respect the privacy of those who will be affected by that software.

3.13 Be careful to use only accurate data derived by ethical and lawful means, and use it only in ways properly authorized.

3.14 Maintain the integrity of data, being sensitive to outdated or flawed occurrences.

3.15 Treat all forms of software maintenance with the same professionalism as new development.

PRINCIPLE 4: JUDGMENT

Software engineers shall maintain integrity and independence in their professional judgment. In particular, software engineers shall, as appropriate:

4.01 Temper all technical judgments by the need to support and maintain human values.

4.02 Only endorse documents either prepared under their supervision or within their areas of competence and with which they are in agreement.

4.03 Maintain professional objectivity with respect to any software or related documents they are asked to evaluate.

4.04 Not engage in deceptive financial practices such as bribery, double billing, or other improper financial practices.

4.05 Disclose to all concerned parties those conflicts of interest that cannot reasonably be avoided or escaped.

4.06 Refuse to participate, as members or advisors, in a private, governmental or professional body concerned with software related issues, in which they, their employers or their clients have undisclosed potential conflicts of interest.

PRINCIPLE 5: MANAGEMENT

Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance. In particular, those managing or leading software engineers shall, as appropriate:

5.01 Ensure good management for any project on which they work, including effective procedures for promotion of quality and reduction of risk.

5.02 Ensure that software engineers are informed of standards before being held to them.

5.03 Ensure that software engineers know the employer's policies and procedures for protecting passwords, files and information that is confidential to the employer or confidential to others.

5.04 Assign work only after taking into account appropriate contributions of education and experience tempered with a desire to further that education and experience.

5.05 Ensure realistic quantitative estimates of cost, scheduling, personnel, quality and outcomes on any project on which they work or propose to work, and provide an uncertainty assessment of these estimates.

5.06 Attract potential software engineers only by a full and accurate description of the conditions of employment.

5.07 Offer fair and just remuneration.

5.08 Not unjustly prevent someone from taking a position for which that person is suitably qualified.

5.09 Ensure that there is a fair agreement concerning ownership of any software, processes, research, writing, or other intellectual property to which a software engineer has contributed.

5.10 Provide for due process in hearing charges of violation of an employer's policy or of this Code.

5.11 Not ask a software engineer to do anything inconsistent with this Code.

5.12 Not punish anyone for expressing ethical concerns about a project.

FIGURE 9.5 Software engineers shall help develop an organizational environment favorable to acting ethically (Clause 6.01).

PRINCIPLE 6: PROFESSION

Software engineers shall advance the integrity and reputation of the profession consistent with the public interest. In particular, software engineers shall, as appropriate:

6.01 Help develop an organizational environment favorable to acting ethically.

6.02 Promote public knowledge of software engineering.

6.03 Extend software engineering knowledge by appropriate participation in professional organizations, meetings and publications.

6.04 Support, as members of a profession, other software engineers striving to follow this Code.

6.05 Not promote their own interest at the expense of the profession, client or employer.

6.06 Obey all laws governing their work, unless, in exceptional circumstances, such compliance is inconsistent with the public interest.

6.07 Be accurate in stating the characteristics of software on which they work, avoiding not only false claims but also claims that might reasonably be supposed to be speculative, vacuous, deceptive, misleading, or doubtful.

6.08 Take responsibility for detecting, correcting, and reporting errors in software and associated documents on which they work.

6.09 Ensure that clients, employers, and supervisors know of the software engineer's commitment to this Code of ethics, and the subsequent ramifications of such commitment.

6.10 Avoid associations with businesses and organizations which are in conflict with this code.

6.11 Recognize that violations of this Code are inconsistent with being a professional software engineer.

6.12 Express concerns to the people involved when significant violations of this Code are detected unless this is impossible, counter-productive, or dangerous.

6.13 Report significant violations of this Code to appropriate authorities when it is clear that consultation with people involved in these significant violations is impossible, counter-productive or dangerous.

PRINCIPLE 7: COLLEAGUES

Software engineers shall be fair to and supportive of their colleagues. In particular, software engineers shall, as appropriate:

7.01 Encourage colleagues to adhere to this Code.

7.02 Assist colleagues in professional development.

7.03 Credit fully the work of others and refrain from taking undue credit.

7.04 Review the work of others in an objective, candid, and properly documented way.

7.05 Give a fair hearing to the opinions, concerns, or complaints of a colleague.

7.06 Assist colleagues in being fully aware of current standard work practices including policies and procedures for protecting passwords, files and other confidential information, and security measures in general.

7.07 Not unfairly intervene in the career of any colleague; however, concern for the employer, the client or public interest may compel software engineers, in good faith, to question the competence of a colleague.

7.08 In situations outside of their own areas of competence, call upon the opinions of other professionals who have competence in that area.

PRINCIPLE 8: SELF

Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession. In particular, software engineers shall continually endeavor to:

8.01 Further their knowledge of developments in the analysis, specification, design, development, maintenance and testing of software and related documents, together with the management of the development process.

8.02 Improve their ability to create safe, reliable, and useful quality software at reasonable cost and within a reasonable time.

8.03 Improve their ability to produce accurate, informative, and well-written documentation.

8.04 Improve their understanding of the software and related documents on which they work and of the environment in which they will be used.

8.05 Improve their knowledge of relevant standards and the law governing the software and related documents on which they work.

FIGURE 9.6 Software engineers shall continually endeavor to improve their ability to create safe, reliable, and useful quality software at reasonable cost and within a reasonable time (Clause 8.02).

8.06 Improve their knowledge of this Code, its interpretation, and its application to their work.

8.07 Not give unfair treatment to anyone because of any irrelevant prejudices.

8.08 Not influence others to undertake any action that involves a breach of this Code.

8.09 Recognize that personal violations of this Code are inconsistent with being a professional software engineer.

9.4 Analysis of the Code

In this section, we analyze the Code and derive an alternate set of underlying principles upon which it rests.

9.4.1 Preamble

The preamble to the Code points out that there is no mechanical process for determining the correct actions to take when faced with a moral problem. Our experience evaluating moral problems related to the introduction and use of information technology confirms this statement. Even two people with similar philosophies may reach different conclusions when confronted with a moral problem. Two Kantians may agree on the basic facts of a moral problem, but disagree on how to characterize the will of the moral agent. Two utilitarians may agree on the benefits and harms resulting from a proposed action, but assign different weights to the outcomes, causing them to reach opposite conclusions.

The preamble also warns against taking an overly legalistic view of the Code. Simply because an action is not expressly forbidden by the Code does not mean it is morally acceptable. Instead, judgment is needed to detect when a moral problem has arisen and to determine the right thing to do in a particular situation.

While the Code is expressed as a collection of rules, these rules are based on principles grounded in different ethical theories. This is not surprising, considering that the Code was drafted by a committee. When we encounter a situation where two rules conflict, the preamble urges us to ask questions that will help us consider the principles underlying the rules. These questions demonstrate the multifaceted grounding of the Code:

1. *Who is affected*?

   Utilitarians focus on determining how an action benefits or harms other people.

2. *Am I treating other human beings with respect*?

   Kant's Categorical Imperative tells us to treat others as ends in themselves, rather than simply as a means to an end.

3. *Would my decision hold up to public scrutiny*?

   A cultural relativist is concerned about whether an action conforms with the mores of society.

4. *How will those who are least empowered be affected*?

   Rawls's second principle of justice requires us to consider whether inequalities are to the greatest benefit of the least-advantaged members of society.

5. *Are my acts worthy of the ideal professional*?

   The ethics of virtue is based on imitation of morally superior role models. Since we did not discuss virtue ethics in Chapter 2, let's examine it now.

9.4.2 Virtue Ethics

ORIGIN OF VIRTUE ETHICS

In *The Nicomachean Ethics*, Aristotle expresses the opinion that happiness results from living a life of virtue [3]. He distinguishes between *intellectual virtue*, which is developed through education, and *moral virtue*, which comes about through repetition of the appropriate acts (Figure 9.7). You can acquire the virtue of honesty, for example, by habitually telling the truth. According to Aristotle, deriving pleasure from a virtuous act is a sign that you have acquired that virtue.

There is a wealth of virtues, of course. Here is a brief list of two dozen virtues given by James Rachels: benevolence, civility, compassion, conscientiousness, cooperativeness, courage, courteousness, dependability, fairness, friendliness, generosity, honesty, industriousness, justice, loyalty, moderation, patience, prudence, reasonableness, self-discipline, self-reliance, tactfulness, thoughtfulness, and tolerance [4].

A person who possesses many moral virtues has a strong moral character. According to Aristotle, when people with strong character face a moral problem, they know the

Virtue ethics avoids the pitfall of impartiality by rejecting the notion that every action must be designed to produce the maximum benefit for people overall [5]. Instead, some virtues are partial toward certain people, while others are impartial and treat everyone equally. Generosity and loyalty are examples of virtues that allow a person to be partial toward friends and family members. Honesty, civility, and courteousness are examples of virtues that a person would extend equally to all human beings.

VIRTUE ETHICS COMPLEMENTS OTHER THEORIES

Rather than treating virtue ethics as a stand-alone theory, some ethicists believe it makes more sense to see virtue ethics as a complement to one of the other theories, such as utilitarianism. Adding virtue ethics allows ethical decision makers to consider their rationale for taking the action as well as the beneficial or harmful effects of the action.

Remember the problem of moral luck, one of the major criticisms of act utilitarianism? Since an action is judged right or wrong based solely on its consequences, an unlucky, unintended consequence can result in an action being considered wrong. Suppose your mother-in-law is in the hospital and you send her an expensive and beautiful bouquet of flowers. Unfortunately, she gets an allergic reaction to one of the flowers in the bouquet. As a result, she must spend an additional four days in the hospital. From a purely act utilitarian point of view, you did the wrong thing when you sent your mother-in-law the flowers. In a mixed act utilitarian/virtue ethics theory, we would also take into account that you were acting out of thoughtfulness, a virtue. If nothing else, introducing the virtue ethics component makes it easier for us to think about some of the other consequences of the action. Despite the allergic reaction, your mother-in-law appreciated your kind gesture, a benefit. In addition, you strengthened your habit of thoughtfulness by practicing it on your mother-in-law, another benefit.

9.4.3 Alternative List of Fundamental Principles

The start of each section of the Code begins with the statement of a fundamental principle. For example, the first section begins with the fundamental principle, "Software engineers shall act consistently with the public interest." All of these statements of fundamental principles are expressed from the point of view of what software engineers ought to do.

Another way to devise a list of fundamental principles is to consider those virtues we would like to instill among all the members of any profession. We end up with a set of general, discipline-independent rules that cut across the eight categories of the Code. Here is an alternative list of fundamental principles derived using that approach:

1. *Be impartial.*

   The good of the general public is equally important to the good of your organization or company. The good of your profession and your company are equally important to your personal good. It is wrong to promote your agenda at the expense of your firm, and it is wrong to promote the interests of your firm at the expense of society. (Supports Clauses 1.02, 1.03, 1.05, 1.07, 3.03, 3.12, 4.01, and 6.05.)

2. *Disclose information that others ought to know.*

   Do not let others come to harm by concealing information from them. Do not make misleading or deceptive statements. Disclose potential conflicts of interest. (Supports Clauses 1.04, 1.06, 2.06, 2.07, 3.01, 4.05, 4.06, 5.05, 5.06, 6.07, 6.08, 6.09, 6.12, and 6.13.)

3. *Respect the rights of others.*

   Do not infringe on the privacy rights, property rights, or intellectual property rights of others. (Supports Clauses 2.02, 2.03, 2.05, and 3.13.)

4. *Treat others justly.*

   Everyone deserves fair wages and appropriate credit for work performed. Do not discriminate against others for attributes unrelated to the job they must do. Do not penalize others for following the Code. (Supports Clauses 5.06, 5.07, 5.08, 5.09, 5.10, 5.11, 5.12, 7.03, 7.04, 7.05, 7.07, and 8.07.)

5. *Take responsibility for your actions and inactions.*

   As a moral agent, you are responsible for the things you do, both good and bad. You may also be responsible for bad things that you allow to happen through your inaction. (Supports Clauses 1.01, 3.04, 3.05, 3.06, 3.07, 3.08, 3.10, 3.11, 3.14, 3.15, 4.02, and 7.08.)

6. *Take responsibility for the actions of those you supervise.*

   Managers are responsible for setting up work assignments and training opportunities to promote quality and reduce risk. They should create effective communication channels with subordinates so that they can monitor the work being done and be aware of any quality or risk issues that arise. (Supports Clauses 5.01, 5.02, 5.03, and 5.04.)

7. *Maintain your integrity.*

   Deliver on your commitments and be loyal to your employer, while obeying the law. Do not ask someone else to do something you would not be willing to do yourself. (Supports Clauses 2.01, 2.04, 2.08, 2.09, 3.01, 3.02, 3.09, 4.03, 4.04, 6.06, 6.10, 6.11, 8.08, and 8.09.)

8. *Continually improve your abilities.*

   Take advantage of opportunities to improve your software engineering skills and your ability to put the Code to use. (Supports Clauses 8.01, 8.02, 8.03, 8.04, 8.05, and 8.06.)

9. *Share your knowledge, expertise, and values.*

   Volunteer your time and skills to worthy causes. Help bring others to your level of knowledge about software engineering and professional ethics. (Supports Clauses 1.08, 6.01, 6.02, 6.03, 6.04, 7.01, 7.02, and 7.06.)

In the following section, we will use these fundamental, discipline-independent principles to facilitate our analysis in four case studies related to computing.

9.5 Case Studies

Throughout this text we have evaluated a wide range of moral problems. Our methodology has been to evaluate the moral problem from the point of view of Kantianism, act utilitarianism, rule utilitarianism, and social contract theory.

Another way to evaluate information technology–related moral problems is to make use of the Software Code of Ethics and Professional Practice. We follow a three-step process:

1. Consult the list of fundamental principles and identify those that are relevant to the moral problem.
2. Search the list of clauses accompanying each of the relevant fundamental principles to see which speak most directly to the issue.
3. Determine whether the contemplated action aligns with or contradicts the statements in the clauses. If the action is in agreement with all of the clauses, that provides strong evidence the action is moral. If the action is in disagreement with all of the clauses, it is safe to say the action is immoral.

   Usually, the contemplated action will be supported by some clauses and opposed by others. When this happens, we must use our judgment to determine which of the clauses are most important before we can reach a conclusion about the morality of the contemplated action.

In the remainder of this section, we will apply this methodology to four case studies.

9.5.1 Software Recommendation

~ Scenario

Sam Shaw calls the Department of Computer Science at East Dakota State University seeking advice on how to improve the security of his business's local area network. A secretary in the department routes Mr. Shaw's call to Professor Jane Smith, an internationally recognized expert in the field. Professor Smith answers several questions posed by Mr. Shaw regarding network security. When Mr. Shaw asks Professor Smith to recommend a software package to identify security problems, Professor Smith tells him that NetCheks got the personal computer magazine's top rating. She does not mention that the same magazine gave a "best buy" rating to another product with fewer features but a much lower price. She also fails to mention that NetCheks is a product of a spin-off company started by one of her former students and that she owns 10 percent of the company.

Analysis

From our list of nine fundamental principles, three are most relevant here:

- Be impartial.
- Disclose information that others ought to know.
- Share your knowledge, expertise, and values.

Searching the list of clauses identified with these fundamental principles, the following ones seem to fit the case study most closely:

- *1.06. Be fair and avoid deception in all statements, particularly public ones, concerning software or related documents, methods and tools.*
  Professor Smith was deceptive when she mentioned the most highly rated software package but not the one rated to be a "best buy."
- *1.08. Be encouraged to volunteer professional skills to good causes and contribute to public education concerning the discipline.*
- *6.02. Promote public knowledge of software engineering.*
  Professor Smith freely provided Sam Shaw with valuable information about network security.
- *4.05. Disclose to all concerned parties those conflicts of interest that cannot reasonably be avoided or escaped.*
- *6.05. Not promote their own interest at the expense of the profession, client or employer.*
  Professor Smith did not tell Sam Shaw that she had a personal stake in the success of the NetCheks software. She did not tell him about the "best buy" package that may have provided him every feature he needed at a much lower price.

Mr. Shaw was asking Professor Smith for free advice, and she provided it. When she freely shared her knowledge about network security, she was acting in the spirit of Clauses 1.08 and 6.02, and doing a good thing.

However, Professor Smith appears to have violated the other three clauses to at least some degree. Most importantly, she did not reveal her personal interest in NetCheks, which could lead her to be biased. The fact that the she did not mention the "best buy" package is evidence that she was neither evenhanded nor completely forthcoming when she answered Mr. Shaw's question about software packages.

Perhaps Mr. Shaw should have heeded the maxim, "Free advice is worth what you pay for it." Nevertheless, the ignorance or foolishness of one person does not excuse the bad behavior of another. Professor Smith should have revealed her conflict of interest. At that point, Mr. Shaw could have chosen to get another opinion if he so desired. ~

9.5.2 Child Pornography

~ SCENARIO

Joe Green, a system administrator for a large corporation, is installing a new software package on the PC used by employee Chuck Dennis. The company has not authorized Joe to read other people's emails, Web logs, or personal files. However, in the course of installing the software, he accidentally comes across directories containing files with suspicious-looking names. He opens a few of

the files and discovers they contain child pornography. Joe believes possessing such images is against federal law. What should he do?

Analysis

Looking over the list of nine fundamental principles, we find these to be most relevant to our scenario:

- Be impartial.
- Respect the rights of others.
- Treat others justly.
- Maintain your integrity.

We examine the lists of clauses associated with these four fundamental principles and identify those which are most relevant:

- *2.03. Use the property of a client or employer only in ways properly authorized, and with the client's or employer's knowledge and consent.*
  Somebody has misused the company's PC by using it to store images of child pornography. By this principle Joe has an obligation to report what he discovered.
- *2.09. Promote no interest adverse to their employer or client, unless a higher ethical concern is being compromised; in that case, inform the employer or another appropriate authority of the ethical concern.*
  While revealing the existence of the child pornography may harm the employee, possessing child pornography is illegal. Applying this principle would lead Joe to disclose what he discovered.
- *3.13. Be careful to use only accurate data derived by ethical and lawful means, and use it only in ways properly authorized.*
  Joe discovered the child pornography by violating the company's policy against examining files on personal computers used by employees.
- *5.10. Provide for due process in hearing charges of violation of an employer's policy or of this Code.*
  Simply because Chuck had these files on his computer does not necessarily mean he is guilty. Perhaps someone else broke into Chuck's computer and stored the images there.

Our analysis is more complicated because Joe violated company policy to uncover the child pornography on Chuck's PC. Once he has this knowledge, however, the remaining principles guide Joe to reveal what he has discovered to the relevant authorities within the corporation, even though management may punish Joe for breaking the privacy policy. There is the possibility that Chuck is a victim. Someone else may be trying to frame Chuck or use his computer as a safe stash for their collection of images. Joe should be discreet until a complete investigation is completed and Chuck has had the opportunity to defend himself.

9.5.3 Anti-Worm

SCENARIO

The Internet is plagued by a new worm that infects PCs by exploiting a security hole in a popular operating system. Tim Smart creates an anti-worm that exploits the same security hole to spread from PC to PC. When Tim's anti-worm gets into a PC, it automatically downloads a software patch that plugs the security hole. In other words, it fixes the PC so that it is no longer vulnerable to attacks via that security hole [6].

Tim releases the anti-worm, taking precautions to ensure that it cannot be traced back to him. The anti-worm quickly spreads throughout the Internet, consuming large amounts of network bandwidth and entering millions of computers. To system administrators, it looks just like another worm, and they battle its spread the same way they fight all other worms [7].

Analysis

These fundamental principles are most relevant to the anti-worm scenario:

- Continually improve your abilities.
- Share your knowledge, expertise, and values.
- Respect the rights of others.
- Take responsibility for your actions and inactions.

Examining the list of clauses associated with each of these fundamental principles reveals those that are most relevant to our case study:

- *1.01. Accept full responsibility for their own work.*
  Tim tried to prevent others from discovering that he was the author of the anti-worm. He did not accept responsibility for what he had done.
- *1.08. Be encouraged to volunteer professional skills to good causes and contribute to public education concerning the discipline.*
  The anti-worm did something good by patching security holes in PCs. Tim provided the anti-worm to the Internet community without charge. However, system administrators spent a lot of time trying to halt the spread of the anti-worm, a harmful effect.
- *2.03. Use the property of a client or employer only in ways properly authorized, and with the client's or the employer's knowledge and consent.*
  Tim's "client" is the community of Internet PC owners who happen to use the operating system with the security hole. While his anti-worm was designed to benefit them, it entered their systems without their knowledge or consent. The anti-worm also consumed a great deal of network bandwidth without the consent of the relevant telecommunications companies.

- *8.01. Further their knowledge of developments in the analysis, specification, design, development, maintenance, and testing of software and related documents, together with the management of the development process.*
- *8.02. Improve their ability to create safe, reliable, and useful quality software at reasonable cost and within a reasonable time.*
- *8.06. Improve their knowledge of this Code, its interpretations, and its application to their work.*
  Tim followed the letter of these three clauses when he acquired a copy of the worm, figured out how it worked, and created a reliable anti-worm in a short period of time. The experience improved his knowledge and skills. Perhaps he should invest some time improving his ability to interpret and use the Code of Ethics!

According to some of these principles, Tim did the right thing. According to others, Tim was wrong to release the anti-worm. How do we resolve this dilemma? We can simplify our analysis by deciding that Tim's welfare is less important than the public good. Using this logic, we will no longer consider the fact that Tim improved his technical knowledge and skills by developing and releasing the anti-worm.

That leaves us with three clauses remaining (1.01, 1.08, and 2.03). From the point of view of Clause 1.01, what Tim did was wrong. By attempting to hide his identity, Tim refused to accept responsibility for launching the anti-worm. He has clearly violated the Code of Ethics in this regard.

When we evaluate Tim's action from the point of view of Clause 1.08, we must determine whether his efforts were directed to a "good cause." Certainly Tim's anti-worm benefited the PCs it infected by removing a security vulnerability. However, it harmed the Internet by consuming large amounts of bandwidth, and it harmed system administrators, who spent time battling it. Because there were harmful as well as beneficial consequences, we cannot say that Tim's efforts were directed to a completely good cause.

Finally, let's evaluate Tim's action from the point of view of Clause 2.03. Even though the anti-worm was completely benevolent, Tim violated the property rights of the PC owners, because the anti-worm infected their PCs without authorization. Hence, Tim's release of the anti-worm was wrong from the point of view of this Clause.

To summarize our analysis, Tim's release of the anti-worm is clearly wrong from the point of view of Clauses 1.01 and 2.03. It is also hard to argue that he satisfied the spirit of Clause 1.08. We conclude that Tim's action violated the Software Engineering Code of Ethics and Professional Practice. ~

9.5.4 Consulting Opportunity

~ Scenario

Acme Corporation licenses a sophisticated software package to many state, county, and city governments. Government agencies have the choice of three

levels of service: the bronze level provides online support only; the silver level adds phone support; and the gold level includes training classes taught on the customer's site. The gold level of support costs $20,000/year more than the silver level.

Jean is one of the Acme employees who works in the support organization. Mostly Jean provides phone support, but from time to time he teaches an on-site class. In fact, Jean created many of the instructional materials used in these classes. Because of the recession, quite a few government agencies have dropped from the gold level of support to the silver level, and some members of Jean's training group have lost their jobs. Jean has a family to support, and he is wondering if his position will soon be eliminated as well.

The state government of East Dakota is one of the many customers that no longer pay Acme Corporation for on-site training. One day Jean gets a call from Maria, who works for the East Dakota state agency using the software package. Maria offers to pay Jean $5,000 plus expenses to run a five-day training class that covers the same material as the official course taught by Acme.

Jean accepts the offer, but he does not inform anyone at Acme Corporation of his decision. Working at home on evenings and weekends, he develops his own set of instructional materials. He takes a week of paid vacation from work, travels to East Dakota, and teaches the class.

Analysis

From our list of fundamental principles, quite a few are relevant here:

- Be impartial.
- Take responsibility for your actions and inactions.
- Disclose information that others ought to know.
- Maintain your integrity.
- Continually improve your abilities.

Examining the clauses associated with each of these fundamental principles, the ones that most closely fit this case study are:

- *3.04 Ensure that they are qualified for any project on which they work or propose to work by an appropriate combination of education and training, and experience.*
  Based on his prior experience at Acme, Jean was certainly well qualified to develop the instructional materials and teach the class in East Dakota. He has fulfilled this obligation of the Code.
- *8.04 Improve their understanding of the software and related documents on which they work and of the environment in which they will be used.*
  By creating his own set of instructional materials, Jean probably developed an even better understanding of the software package and its capabilities. There is a good chance he came up with some insights about better ways to

teach others how to use the software. This additional knowledge will make Jean a more valuable employee of Acme Corporation.

- *4.05 Disclose to all concerned parties those conflicts of interest that cannot reasonably be avoided or escaped.*
  By accepting the consulting job with the East Dakota state government, Jean created a conflict of interest between himself and Acme Corporation. Namely, it is in Jean's interest if East Dakota does not purchase the gold level of support, but it is in Acme Corporation's interest if East Dakota does buy the gold level of support. Jean violated this clause by not disclosing his consulting job to Acme Corporation.
- *2.08 Accept no outside work detrimental to the work they perform for their primary employer.*
  Employers provide employees with weekends off and paid vacations so that they can rest from their labors and return to work refreshed and able to perform at a high level. You could argue that Jean's consulting work was detrimental to his "day job" at Acme Corporation because it filled his evenings and weekends and kept him from getting a proper vacation.
- *6.05 Not promote their own interest at the expense of the profession, client or employer.*
  By agreeing to teach the class in East Dakota, Jean put his own interest above that of his employer. Clearly the East Dakota state government recognized a need to have some on-site training. If Jean did not accept the consulting job, the East Dakota government may have gone back to the gold level of support from Acme.

You could argue that Jean is actually helping Acme Corporation. Governments are dropping the gold level of support because it is simply too expensive, but phone support and online support aren't enough. If these agencies cannot find another source of on-site training, they may stop using Acme's software altogether. By providing East Dakota with affordable, on-site training, Jean was helping ensure that East Dakota would remain a customer of Acme Corporation, albeit at the silver level.

You could also argue that Jean's work for East Dakota improved his knowledge of the software package and his ability to teach others how to use it, making him a more effective phone support person at Acme.

However, it's unlikely upper management at Acme Corporation will be convinced by these arguments, particularly since Jean did not disclose the offer from East Dakota before accepting it. Jean's decision is much more likely to cause management to question his loyalty to his company and his fellow employees. If the company learns about his consulting work, Jean may well be the next person laid off.

To conclude our analysis, Jean's actions were wrong and unwise. He violated clauses 2.08, 4.05, and 6.05 of the Software Engineering Code of Ethics and Professional Practice, and he may have put his full-time job in jeopardy. ~