An Introduction to Software Engineering Ethics

MODULE AUTHORS:
Shannon Vallor, Ph.D. Associate Professor of Philosophy, Santa Clara University
SPECIAL CONTRIBUTOR TO INTRODUCTION:
Arvind Narayanan, Ph.D. Assistant Professor of Computer Science, Princeton University

These documents contain fields that can be filled in by users who have downloaded the free Adobe Reader. Simply download the appropriate document to your computer, type your comments in the boxes, and save the completed version. To send the version with your responses, include it as an attachment to an email.

What do we mean when we talk about 'ethics'?

Ethics in the broadest sense refers to the concern that humans have always had for figuring out *how best to live.* The philosopher Socrates is quoted as saying in 399 B.C., "the most important thing is not life, but the good life."[1] We would all like to avoid a life that is shameful and sad, wholly lacking in achievement, love, kindness, beauty, pleasure or grace. Yet what is the best way to achieve the opposite of this – a life that is not only acceptable, but even excellent and worthy of admiration? This is the question that the study of ethics attempts to answer.

Today, the study of ethics can be found in many different places. As an academic field of `study, it belongs primarily to the discipline of philosophy, where scholars teach and publish research about the nature and structure of ethical norms. In community life, ethics is pursued through diverse cultural, political and religious ideals and practices. On a personal level, it can be expressed in an individual's self-reflection and continual strivings to become a better person. In work life, it is often formulated in formal codes or standards to which all members of a profession are held, such as those of medical ethics. Professional ethics is also taught in dedicated courses, such as business ethics. It can also be infused into courses such as this one.

What is ethics doing in a course for software engineers?

Like medical, legal and business ethics, **engineering ethics** is a well-developed area of professional ethics in the modern West. The first codes of engineering ethics were formally adopted by American engineering societies in 1912-1914. In 1946 the National Society of Professional Engineers (NSPE) adopted their first formal Canons of Ethics. In 2000 ABET, the organization that accredits university programs and degrees in engineering, began to formally require the study of engineering ethics in all accredited programs: "Engineering programs must demonstrate that their graduates have an understanding of professional and ethical responsibility."[2] Professional engineers today, then, are expected to both *learn about* and *live up to* ethical standards as a condition of their membership in the profession.

---

[1] Plato, *Crito* 48b. In Cahn (2010).
[2] ABET 2000 criterion 3(f) (ABET, 1998).

But the average computer/software engineering student might still be confused about how and why this requirement should apply to them. Software engineering is a relatively young practice and compared with other engineering disciplines, its culture of professionalism is still developing. This is reinforced by the fact that most engineering ethics textbooks focus primarily on ethical issues faced by civil, mechanical or elecrical engineers. The classic case studies of engineering ethics depict catastrophic losses of life or injury as a result of ethical lapses in these fields: the Challenger explosion, the Ford Pinto fires, the Union Carbide/Bhopal disaster, the collapse of the Hyatt walkway in Kansas City. When we think about the engineer's most basic ethical duty to "hold paramount the safety, health, and welfare of the public,"[3] it is clear why these cases are chosen - they powerfully illustrate the importance of an engineer's ethical obligations, and the potentially devastating consequences of failing to live up to them.

But software engineers build *lines of code*, not cars, rockets or bridges full of vulnerable human beings. Where is the comparison here? Well, one answer might already have occurred to you. How many cars or rockets are made today that do *not* depend upon critical software for their safe operation? How many bridges are built today without the use of sophisticated computer programs to calculate expected load, geophysical strain, material strength and design resilience? A failure of these critical software systems can result in death or grievous injury just as easily as a missing bolt or a poorly designed gas tank. This by itself is more than enough reason for software engineers to take seriously the ethics of their professional lives. Is it the *only* reason? What might be some others? Consider the following:

The software development and deployment process in the Internet era has some peculiarities that make the ethical issues for software engineers even *more* acute in some ways than for other types of engineers. First, the shortened lifecycle has weakened and in some cases obliterated software review by management and legal teams. In the extreme, for Web applications like Facebook, it is normal for individual engineers or small groups of engineers to code and deploy features directly, and indeed the culture takes pride in this. Even where more traditional development practices prevail, at least some deployments like bug fixes are shipped with only technical (and not ethical) oversight. At any rate, engineers at least retain the *ability* to deploy code directly to end users, an ability that can easily be abused.

All of this is in stark contrast to say, a civil engineering project with a years-long (or decades-long) lifecycle and multiple layers of oversight. Nor does such a project offer a malicious engineer any real means to obfuscate her output to sneak past standards and safety checks.

Second is the issue of scale, perhaps the defining feature of the software revolution. Typically the entire world is part of the addressable market. Of course, it is scale that has led to the potential for individual engineers to create great good, but with it naturally comes the ability to cause great harm, especially when combined with the first factor above.

---

[3] NSPE Code of Ethics for Engineers, First Fundamental Canon.

Here's a rather benign but illustrative example. On June 9, 2011, Google released a "doodle" honoring Les Paul which users found addictive to play with. This is a type of project that's typically done by an individual engineer on their "20% time" in a day or two. A third party, RescueTime, estimated that 5.3 million hours were spent playing this game.[4] Let us pause to consider that 5.3 million hours equates to about *eight lifetimes*.

Did the doodle make a positive contribution to the world? Do engineers at Google have an obligation to consider this question before releasing the feature? What principle(s) should they use to determine the answer? These are all valid questions, but what is perhaps even more interesting here is the disproportionality between the amount of time engineers spent creating the feature (at most a few person-days, in all likelihood), and the amount of time users spent on it (several lifetimes). Often, in today's world, engineers must grapple with these questions instead of relying on management or anyone else.

Finally, the lack of geographic constraints means that engineers are generally culturally unfamiliar with some or most of their users. The cost-cutting imperative often leaves little room for user studies or consultations with experts that would allow software development firms to acquire this familiarity. This leads to the potential for privacy violations, cultural offenses, and other such types of harm.

For example, people in many countries are notoriously sensitive to the representation of disputed border territories on maps. In one recent example, an error in Google maps led to Nicaragua dispatching forces to its border with Costa Rica. Google then worked with US State Department officials to correct the error.[5]

On top of these considerations, software engineers share with everyone a basic human desire to flourish and *do well* in life and work. What does that have to do with ethics? Imagine a future where you are faced with a moral quandary arising from a project you are working on that presents serious risks to users. In that scenario, will you act in a way that you would be comfortable with if it later became public knowledge? Would it matter to you whether your family was proud or shamed by your publicly exposed actions? Would it matter to you whether, looking back, you saw this as one of your *better* moments as a human being, or one of your *worst*? Could you trust anyone to whom these outcomes *didn't* matter?

Thus ethical obligations have both a *professional* and a *personal* dimension. Each are essential to consider; without a sense of personal ethics, one would be indifferent to their effect on the lives of others in circumstances where one's professional code is silent. To understand what's dangerous about this, consider any case in human history when a perpetrator of some grossly negligent, immoral or inhumane conduct tries to evade their responsibility by saying, 'I was just following orders!' So personal ethics helps us to be sure that we take full responsibility for our moral choices and their consequences.

---

[4] "Google Doodle Strikes Again! 5.3 Million Hours Strummed," Rescue Time, Jun 9 2011.

[5] "Google Maps Embroiled in Central America Border Dispute," AFP, Nov 6 2010.

But for professionals who serve the public or whose work impacts public welfare, a personal code of ethics is just not enough. Without a sense of professional ethics, one might be tempted to justify conduct in one's own mind that could never be justified in front of others. Additionally, professional ethics is where one learns to see how broader ethical standards/values (like honesty, integrity, compassion and fairness) apply to one's particular type of work. For example, wanting to have integrity is great – but what does integrity look like in a software engineer? What sort of specific coding practices demonstrate integrity, or a lack of it? This is something that professional codes of ethics can help us learn to see. Finally, being a professional means being a part of a moral community of others who share the same profound responsibilities we do. We can draw strength, courage, and wisdom from those members of our professional community who have navigated the same types of moral dilemmas, struggled with the same sorts of tough decisions, faced up to the same types of consequences, and ultimately earned the respect and admiration of their peers and the public.

Broadening our view of software engineering ethics

Certainly, software engineers must concern themselves primarily with the health, safety and welfare of those who are affected by their work, as the so-called 'paramountcy clause' of NSPE's Code of Ethics states. But we need to broaden our understanding of a number of aspects of this claim, including:

- The **types** of harms the public can suffer as result of this work;
- How software engineers **contribute** to the good life for others;
- **Who** exactly are the 'public' to whom the engineer is obligated;
- **Why** the software engineer is obligated to protect the public;
- What **other** ethical obligations software engineers are under;
- **How** software engineers can actually live up to ethical standards;
- What is the end **goal** of an ethical life in software engineering;
- What are the professional **codes** of software engineering ethics;

Let's begin with the first point.

PART ONE

What kinds of harm to the public can software engineers cause? What kinds of harm can they prevent?

We noted above that failures of critical software systems can result in catastrophic loss of life or injury to the public. If such failures result, directly or indirectly, from software engineers' choices to ignore their professional obligations, then these harms are clearly the consequences of *unethical* professional behavior. Those responsible each bear the moral weight of this avoidable human suffering, whether or not this also results in legal, criminal or professional punishment.

But what *other* kinds of harms do software engineers have an ethical duty to consider, and to try to prevent? Consider the following scenario:

Case Study 1

Mike is a father of 3, and in order to save for their college educations, he has been working two jobs since his kids were born. His daughter Sarah has worked as hard as she can in high school to get high grades and SAT scores; as a result of her hard work she has been accepted to a prestigious Ivy-League college, and the deposit for her first year is due today. If the deposit goes unpaid, Sarah loses her spot in the freshman class. Mike paid the bill last week, but today he gets an email from the college admissions office saying that his payment was rejected for insufficient funds by his bank, and if he does not make the payment by the end of the day, Sarah will lose her place and be unable to attend in the Fall. Panicked, Mike calls the bank – he had more than enough money in his savings to cover the bill, so he cannot understand what has happened. The bank confirms that his account had plenty of funds the day before, but cannot tell him why the funds are gone now or why the payment was rejected. They tell him there must be some 'software glitch' involved and that they will open an investigation, but that it will take weeks to resolve. They will only restore the funds in his account once the investigation is completed and the cause found. Mike has no other way to get the money for the deposit on such short notice, and has to tell Sarah that he couldn't cover the bill despite his earlier promise, and that she won't be attending college in the Fall.

Question 1.1:

What kinds of harm has Mike probably suffered as a result of this incident? What kinds of harm has Sarah probably suffered? (Make your answers as full as possible; identify as many kinds of harm done as you can think of).

Type your answer here

Question 1.2:

Could the problem with Mike's account have been the result of an action (or a *failure* to perform an action) by a software engineer? How many possible scenarios/explanations for this event can you think of that involve the conduct of one or more software engineers? Briefly explain the scenarios:

Type your answer here (continue answer on next page if needed)

(Continue your answer to 1.2 from previous page)

Question 1.3:

Taking into account what we said about ethics in the introduction, could any of the scenarios you imagined involve an *ethical* failure of the engineer(s) responsible? How? Explain:

***Note: An ethical failure would be preventable, and one that a good human being with appropriate professional care and concern *would* and *should* have prevented (or at least have made a serious effort to prevent).**

Type your answer here

Let's try a different scenario:

Case Study 2

Karen is a young lawyer at a prestigious firm with an incredibly hectic and stressful schedule, who needs to organize what little free time she has more efficiently. She has just downloaded a new app called Errand Whiz onto her iPhone; this app merges information from Karen's to-do list, information on her purchasing habits from retail stores she shops at, and GPS software to produce the most efficient map and directions for running errands on her days off. Based on what it knows about what she needs to purchase and her general shopping habits, it tells Karen what locations of her favorite stores to visit on a given day, in what order and by what routes – this way she can get her errands done in the least amount of time, traveling the least number of miles. To accomplish this, the app aggregates information not only about where she lives and shops, but also tracks what she typically buys in each store, how much she buys, what she typically pays for each item. This collected data is not stored on Karen's phone, but on a separate server that the app links to when it needs to create a shopping map. The app encourages users to log in via Facebook, as the developers have made a deal with Facebook to sell this data to third-party advertisers, for the purpose of targeting Facebook ads to Karen and her friends.

Question 1.4: In what ways could Karen potentially be harmed by this app, depending on how it is designed and how her shopping data is handled and used? Identify a few harmful scenarios you can think of, and the *types* of harm she could suffer in each:

Type your answer here (continue on next page if needed)

(Continue your answer to 1.4 from previous page)

Question 1.5: Which if any of these harms could result from ethical failings on the part of the people who developed Errand Whiz? *How*, specifically?

Type your answer here (continue on next page if needed)

(Continue your answer to 1.5 from previous page)

Question 1.6: What actions could the people behind Errand Whiz take to prevent these harms? Are they ethically *obligated* to prevent them? Why or why not? Explain your answer.

Type your answer here

Ideally, these scenarios have helped to broaden your understanding of the ethical scope of software engineering. In considering and protecting the 'health, safety and welfare' of the public, we must not limit our thinking to those contexts in which our design choices or coding practices have the potential to cause someone's death, or cause them direct physical injury. *The harms that people can suffer as a result of failures by software engineers to consider their ethical obligation to the public are far more numerous and more complex than we might think.*

PART TWO

How do software engineers contribute to the good life for others?

There is a second way in which we need to broaden our understanding of engineering ethics. Ethics is not just about avoiding harms, as a narrow focus on preventing catastrophic events might make us believe. Ethics is just as much about doing *good*. 'Doing good' is not something that matters only to missionaries, social workers and philanthropists. To live a 'good life' is to make a positive contribution to the world through your existence, to be able to say at the end of your life that in your short time here, you made the world at least somewhat *better* than it would have been without you in it. This is also how we think about the lives of those who have left us: when we mourn our friends and loved ones, we comfort ourselves by remembering the unique comforts and joys they brought to our lives, and the lives of others; we remember the creative work they left behind, the problems they helped us solve, and the beautiful acts they performed, great and small. *Could a life about which these things could <u>not</u> be said still be a good life?*

If the good life requires making a positive contribution to the world in which others live, then it would be perverse if we accomplished none of that in our professional lives, where we spend many or most of our waking hours, and to which we devote a large proportion of our intellectual and creative energies. Excellent doctors contribute *health* and *vitality* to their patients and medical *knowledge* to their interns and colleagues; excellent professors cultivate *knowledge*, *insight*, *skill and confidence* in their students and contribute the benefits of their *research* to the wider community; excellent lawyers contribute *balance*, *fairness* and *intellectual vigor* to a larger system of justice.

Question 2.1: What sorts of things can excellent *software engineers* contribute to the good life?

(Answer as fully/in as many ways as you are able):

Question 2.2: What kinds of character traits, qualities, behaviors and/or habits do you think mark the kinds of people who tend to contribute most in these ways?

(Answer as fully/in as many ways as you are able):

PART THREE

To *whom* are software engineers obligated by their professional ethics? Who is 'the public' that deserves an engineer's professional concern?

The NSPE's **paramountcy clause** asks engineers to recognize that their primary professional duty is to 'hold paramount the safety, health and welfare of the public.' But *who* exactly is this 'public?' Of course, one can respond simply with, 'the public is everyone.' But the public is not an undifferentiated mass; the public is composed of our families, our friends and co-workers, our employers, our neighbors, our church or other local community members, our countrymen and women, and people living in every other part of the world. To say that we have ethical obligations to 'everyone' is to tell us very little about *how* to actually work responsibly as an engineer in the public interest, since each of these groups and individuals that make up the public are in a unique relationship to us and our work, and are potentially impacted by it in very different ways. We also have special obligations to some members of the public (our children, our employer, our friends, our fellow citizens) that exist alongside the broader, more general obligations we have to all of them.

One concept that ethicists often use to clarify our obligations to the public is that of a **stakeholder**. A stakeholder is anyone who is potentially impacted by my actions. Clearly, certain persons have *more* at stake than other stakeholders in any given action I might take; when I consider, for example, how much effort to put into cleaning up a buggy line of code in a program that will be used to control a pacemaker, it is obvious that the patients in whom the pacemakers with this programming will be implanted are the primary stakeholders in my action; their very lives are potentially at risk in my choice. And this stake is so ethically significant that it is hard to see how any other stakeholder's interest could weigh as heavily.

Still, in most ethical contexts, including those that arise in software engineering, there are a variety of stakeholders potentially impacted by my action, and their interests may not always align with each other. For example, my employer's interests in cost-cutting and an on-time product delivery schedule may frequently be in tension with the interest of other stakeholders in having the highest quality and most reliable product. Yet even these stakeholder conflicts are rarely so simple as they might first appear; the consumer also has an interest in an affordable product, and my employer also has an interest in earning a reputation for product excellence, and in maintaining the profile of a responsible corporate citizen.

Of course, while my own trivial, short-sighted and self-defeating interests (say, in gaining extra leisure time by taking reckless coding shortcuts) will never trump a critical moral interest of another stakeholder (say, their interest in not being unjustly killed by my product), it remains true that I myself am a stakeholder, since my actions also impact my own life and well-being. A decision to ignore my well-defined contractual obligations to my employer, or my obligations to my fellow product team members, will have weighty consequences for me. But ignoring the health, safety and welfare of those who rely upon the code I produce has consequences that are potentially even graver – for me as well as for those persons whose well-being I have chosen to discount or ignore.

Ethical decision-making thus requires cultivating the habit of reflecting carefully upon the range of stakeholders who together make up the 'public' to whom I am obligated, and weighing what is at stake for each of us in my choice.

Here is a scenario to help you think about what this reflection process can entail:

Case Study *3*

> You are a new hire in a product design team for a start-up company that is developing new and more powerful versions of the kind of packet-sniffing and email scanning software systems used by law enforcement agencies and large corporations to monitor data traffic for illegal activities. This kind of software might, for example, be programmed to detect illegal downloads of copyrighted materials, or to flag for review email keywords like 'bomb,' 'steal,' or 'bribe.' You are a young parent of two small children, with parents and friends who are deeply proud of your achievements. You are looking

forward to using this first job to cultivate a reputation in your industry for being an excellent software engineer.

One day, you happen to overhear your supervisor chatting with another supervisor about a new contract the company has recently received from a foreign government. You happen to recognize the name of this country as one that is currently run by an oppressive military regime that routinely imprisons its citizens without trial or other due process. In this country, people perceived as political dissidents and their families are often sent to labor camps with deplorable living conditions, without hope of appeal, for an indefinite period. Your own nation has strongly criticized this country's human rights record, and many international organizations as well as the United Nations have condemned its practices.

You realize now that the product your team is working on is part of your company's contract with this government; and in fact, you have been assigned specifically to develop the part of the product that searches for specific keyword strings in private emails, texts, social networking messages, Skype and phone conversations. Reviewing the specs for your task, you realize that your contribution to the product will almost certainly be used to identify for extraction and review conversations between private citizens of this country in which there is *any* specific discussion of their government or its policies, and especially those in which words like 'reform,' 'injustice,' 'corruption,' 'due process' or 'human rights' occur.

Question 3.1: Who are the various stakeholders in this scenario, and what do they each have at stake in your action? ***Reflect carefully and deeply, and answer as fully as possible.***

Type your answer here

Question *3.2*: What do you think is your ethical obligation in this situation? What do you think an *excellent* software engineer would do in this situation? Are they the same thing, or different? Please explain your answer.

Type your answer here

PART FOUR

***Why* do software engineers have ethical obligations to the public at all? Where do these obligations come from?**

As you might expect by now, there is a simple answer to this question that will nevertheless lead us into a far more complex and profound set of considerations. The simple answer is, 'because software engineers are human beings, and *all* human beings have ethical obligations to each other.' Unless you believe, for example, that you have no ethical obligation to stop a small toddler who you happen to see crawling toward the opening to a deep mineshaft[6], then you accept that you have some basic ethical obligations toward other human beings.

What those obligations *are*, precisely, is a matter of **ethical theory**, and many such theories have been developed over the course of human history. Some of these theories developed in folk or religious traditions, others are articulated in scholarly philosophical discourse from the ancient world to today. Among the most well-known and influential types of theory are those of **virtue ethics** found in diverse cultures from Confucian ethics to ancient Greek, Roman and Christian philosophy, along with the **consequentialist** group of theories that include *utilitarianism*, and finally **deontological** theories of ethics that emphasize rules and principles. We will briefly revisit these types of ethical theory in the next section.

Our question here, however, was not '*what are my ethical duties?*' but rather '*why do I have them?*' That is not a question for ethical theory, it is a question of **metaethics**, or the study of where our ethical duties come from and why they obligate us to act as they say we should. Many answers have been given to this question, but before we get lost in a profound philosophical problem, let us remember that in our case we are exploring the special ethical obligations of *software engineers*, which while not wholly independent of our broader ethical obligations as human beings, may have a more clearly identifiable source and justification.

The first explanation of this source involves the concept of a **profession**. What is a professional? You may not have considered that this word is etymologically connected with the English verb 'to profess.' What is it to profess something? It is to stand publicly for something, to express a belief, conviction, value or promise to a general audience that you expect that audience to *hold you accountable* for, and to identify you with. When I profess something, I assert that this is something about which I am serious and sincere; and which I want others to know about me. So when we identify someone as a *professional* X (whether 'X' is a lawyer, physician, soldier or engineer), we are saying that being an 'X' is not just a job, but a *vocation*, a form of work to which the individual is committed and with which they would like their lives to be identified.

---

[6] This example is adapted from one given by the Confucian philosopher Mencius (in Ivanhoe and Van Norden 2001), who argued that anyone who would be unmoved by the child's peril was not truly human. In the contemporary medical vernacular we would more likely diagnose such a person as a *psychopath* or *sociopath*.

This is part of why professionals are generally expected to undertake advanced education and training in their field; not only because they need the expertise (though that too), but also because this is a important sign of their investment and commitment to the field. When students who have completed an arduous degree program enter the work world, this is taken as evidence that they are sincere in their interest in *this* kind of work, that they understand and uniquely value the contribution that *this* work makes to the world, and that they want their own personal good and sense of self to be enduringly intertwined and identified with the good of their chosen profession. Of course, people do change professions – but not as frequently, or as lightly, as people change mere *jobs*.

So what does being a professional have to do with ethics? How does it create special ethical obligations for the software engineer? First, I stated above that a professional has stated implicitly a desire to have their own good intertwined and identified with the good of their profession. But what *is* the good of their profession? Look back at your answers to Question 2.1 above. Do they suggest an answer to this question?

Consider that members of most professions enjoy an elevated status in their communities; doctors, professors, scientists and lawyers generally get more respect from the public (rightly or wrongly) than retail clerks, manicurists, toll booth operators, and car salespeople. But why? It can't just be the difference in skill; after all, car salespeople have to have very specialized skills in order to thrive in their job. The distinction lies in the perception that professionals secure a ***vital* public good**, not something of merely private and conditional value. For example, without doctors, public health would certainly suffer – and a good life is virtually impossible without some measure of health. Without lawyers and judges, the public would have no formal access to justice – and without recourse for injustice done to you or others, how can the good life be secure? Without scientists, the public would be deprived of reliable and carefully tested knowledge – and how can a life filled with ignorance and error be good? So each of these professions is supported and respected by the public precisely because they deliver something *vital* to the good life, and something needed not just by a few, but by us all.

We are nearing the conclusion of our inquiry in this section. We started with the question of why we have ethical obligations as software engineers. Well, software engineering is a profession, and one that like all professions, receives distinctive public support and respect. What do software engineers do to *earn* that respect? How must they act in order to *continue* to earn it? After all, special public respect and support are not given for free or given unconditionally – they are given in recognition of some service or value that actually warrants support and respect. That support and respect is also something that translates into real *power*; the power of public funding and consumer loyalty, the power of influence over how people live and what systems they use to organize their lives; in short, the power to guide the course of other human beings' technological future. And as we are told in the popular *Spiderman* saga, "With great power comes great responsibility." This is a further reason, even above their general ethical obligations as human beings, that software engineers have special ethical obligations to the public they serve.

Question 4.1: Which of the contributions in your answer to 2.1 are related to vital public good(s)? What vital public goods might software engineers help to secure that you did not mention in your initial answer to 2.1?

Type your answer here

PART FIVE

What *general* ethical obligations are software engineers under, beyond their distinctive professional obligations?

We noted above that software engineers, in addition to their special professional obligations to the public, also have the same ethical obligations to their fellow human beings that we all share. What might those obligations be, and how should they be evaluated alongside our professional obligations? There are a number of familiar concepts that we already use to talk about how, in general, we ought to treat others. Among them are the concepts of *rights*, *justice* and the *common good*. But how do we define the concrete meaning of these important ideals? Here are three common frameworks for understanding our general ethical duties to others:

Virtue Ethics: Virtue approaches to ethics are found in the ancient Greek and Roman traditions, in Confucian, Buddhist and Christian moral philosophies, and in modern secular thinkers like Hume and Nietzsche. Virtue ethics focuses not on rules for good or bad actions, but on the qualities of morally excellent persons (e.g., virtues). Such theories are said to be *character* based, insofar as they tell us what a person of virtuous character is like, and how that moral character develops. Such theories also focus on the habits of action of virtuous persons, such as the habit of moderation (finding the 'golden mean' between extremes), as well as the virtue of prudence or practical wisdom (the ability to see what is morally required even in new or unusual situations to which conventional moral rules do not apply).

How can virtue ethics help us to understand what our moral obligations are? It can do so in three ways. The first is by helping to see that we have a basic moral obligation to make a consistent and conscious effort to *develop* our moral character for the better; as the philosopher Confucius said, the real ethical failing is not having faults, 'but rather failing to amend them.'[7] The second thing virtue theories can tell us is where to look for standards of conduct to follow; virtue theories tell us to look for them in our own societies, in those special persons who *are* exemplary human beings with qualities of character (virtues) to which we should aspire. The third thing that virtue ethics does is direct us toward the lifelong cultivation of *practical wisdom* or good moral judgment: the ability to discern which of our obligations are most important in a given situation *and* which actions are most likely to succeed in helping us to meet those obligations. Virtuous persons with this ability flourish in their own lives *by* acting justly with others, and contribute to the common good by providing a moral example for others to admire and follow.

[7] Confucius, *Analects.* In Ivanhoe and Van Norden (2001).

Question 5.1: How would a conscious habit of thinking about how to be a better human being contribute to a person's character, especially over time?

Type answers here

Question 5:2: Do you *know* what specific aspects of your character you would need to work on/improve in order to become a better person? (Yes or No)

Type answers here

Question 5:3: Do you think most people make *enough* of a regular effort to work on their character or amend their shortcomings? Do you think we are morally *obligated* to make the effort to become better people? Why or why not?

Type answers here

Question 5:4: Who do you consider a model of moral excellence that you see as an example of how to live, and whose qualities of character you would like to cultivate? Who would you want your children (or future children) to see as examples of such human (and especially moral) excellence?

Type answers here

Consequentialist/Utilitarian Ethics: Consequentialist theories of ethics derive principles to guide moral action from the likely **consequences** of those actions. The most famous form of consequentialism is *utilitarian* ethics, which uses the principle of the 'greatest good' to determine what our moral obligations are in any given situation.[8] The 'good' in utilitarian ethics is measured in terms of happiness or pleasure (where this means not just physical pleasure but also emotional and intellectual pleasures). The absence of pain (whether physical, emotional, etc.) is also considered good, unless the pain somehow leads to a net benefit in pleasure, or prevents greater pains (so the pain of exercise would be good because it also promotes great pleasure as well as health, which in turn prevents more suffering). When I ask what action would promote the 'greater good,' then, I am asking which action would produce, in the long run, the greatest *net sum* of good (pleasure and absence of pain), taking into account the consequences for all those affected by my action (not just myself). This is known as the *hedonic calculus*, where I try to maximize the overall happiness produced in the world by my action.

Utilitarian thinkers believe that at any given time, whichever action among those available to me is most likely to boost the overall sum of happiness in the world is the right action to take, and my moral obligation. This is yet another way of thinking about the 'common good.' But utilitarians are sometimes charged with ignoring the requirements of individual rights and justice; after all, wouldn't a good utilitarian willingly commit a great injustice against one innocent person as long as it brought a greater overall benefit to others? Many utilitarians, however, believe that a society in which individual rights and justice are given the highest importance just *is* the kind of society most likely to maximize overall happiness in the long run. After all, how many societies that deny individual rights, and freely sacrifice individuals/minorities for the good of the many, would we call *happy*?

Question 5:5: What would be the hardest part of living by the utilitarian principle of the 'greatest good'? What would be the most rewarding part?

Type answer here

[8] John Stuart Mill, *Utilitarianism* (1861). In Shafer-Landau (2007).

(Continue your answer to 5.5 from previous page)

Question 5:6: What different kinds of pleasure/happiness are there? Are some pleasures more or less valuable or of higher or lower *quality* than others? Why or why not? Explain your intuitions about this:

Type answer here

Question 5:7: Utilitarians think that pleasure and the absence of pain are the highest goods that we can seek in life, and that we should always be seeking to produce these goods for others (and for ourselves). They claim that every other good thing in life is valued simply because it produces pleasure or reduces pain. Do you agree? Why or why not?

Type answer here

Question 5:8: A utilitarian might say that to measure a 'good life,' you should ask: 'how much overall happiness did this life bring into the world?' Do you agree that this is the correct measure of a good life, or not? Briefly explain.

Type answer here

Deontological Ethics: Deontological ethics are *rule or principle-based* systems of ethics, in which one or more rules/principles are claimed to tell us what our moral obligations are in life. In Judeo-Christian thought, the Ten Commandments can be thought of as a deontological system. Among modern, secular forms of ethics, many deontological systems focus on lists of 'rights' (for example, the rights not to be unjustly killed, enslaved, or deprived of your property). Consider also the modern idea of 'universal human rights' that all countries must agree to respect. In the West, moral rights are often taken as a basis for law, and are often invoked to justify the making of new laws, or the revision or abolition of existing ones. In many cultures of East Asia, deontological systems may focus not on on rights but on *duties*; these are fixed obligations to others (parents, siblings, rulers, fellow citizens etc.) that must be fulfilled according to established rules of conduct that govern various types of human relationships.

Another well-known deontological system is that of the 18th century philosopher Immanuel Kant, who identified a single moral rule called the **categorical imperative**.[9] This principle tells us to only act in ways that we would be willing to have all other persons follow, all of the time. He related this to another principle that tells us never to treat a human being as a 'mere means to an end,' that is, as an object to be manipulated for our own purposes. For example, I might want to tell a lie to get myself out of trouble in a particular case. But I certainly would not want *everyone* in the world to lie every time they felt like it would help *them* avoid trouble. And if someone lies to me to get me to do something that benefits them, I am rightly upset about being treated as a mere object to be manipulated for gain. So I cannot logically give myself permission to lie, since there is nothing about me that exempts me from my *own* general moral standards for human behavior. For if I am willing to give myself permission to act in this way for this reason, how could I logically justify withholding the same permission from others?

According to this principle, human lives are the ultimate sources of all moral value. I thus have a universal moral obligation to treat other human lives in ways that acknowledge and respect their unconditional value, and to not treat them merely as tools to manipulate for lesser purposes. And since I myself am human, I cannot morally allow even *my own* existence to be used as a mere tool for some lesser purpose (for example, to knowingly sell out my personal integrity for money, fame or approval). This principle highlights my duty to always respect the *dignity* of all human lives. This theory is also linked with a particular idea of *justice*, as treatment that recognizes the basic equality and irreplaceable dignity of every human being, no matter who they are or where they live. Such thinking is often considered to be at the heart of the modern doctrine of inalienable human rights.

[9] Immanuel Kant, *Groundwork for the Metaphysics of Morals* (1785). In Shafer-Landau (2007).

Question 5:9: How often, when making decisions, do you consider whether you would willingly permit everyone else to act in the same way that you are choosing to act?

Type answer here

Question 5:10: What are two other examples you can think of, beyond those given in the text above, in which someone is treated as a 'mere means to an end'?

Example 1:

Example 2:

Question 5:11: Do you agree that human lives are of the highest possible value and beyond any fixed 'price'? In your opinion, how well does our society today reflect this view on morality and justice? *Should* it reflect this view?

Type answer here

Question 5:12: While each of the 3 distinct types of ethical frameworks/theories reviewed in this section is subject to certain limitations or criticisms, what aspects of the good life/ethics do you think each one captures *best*?

Type answer here

PART SIX

***How* can software engineers live up to their ethical obligations, both professionally and in their private lives?**

There are a number of common habits and practices that create obstacles to living well in the moral sense; fortunately, there are also a number of common habits and practices that are highly conducive to living well:

Five Ethically Constructive Habits of Mind and Action:

1. Self- Reflection/Examination: This involves spending time on a regular basis (even daily) thinking about the person you want to become, in relation to the person you are today. It involves identifying character traits and habits that you would like to change or improve in your private *and* professional life; reflecting on whether you would be happy if those whom you admire and respect most knew all that *you* know about your actions, choices and character; and asking yourself how fully you are living up to the values you profess to yourself and others.

2. Looking for Moral Exemplars: Many of us spend a great deal of our time, often more than we realize, judging the shortcomings of others. We wallow in irritation or anger at what we perceive as unfair, unkind or incompetent behavior of others, we comfort ourselves by noting the even greater professional or private failings of others, and we justify ignoring the need for our own ethical improvement by noting that many others seem to be in no hurry to become better people either. What we miss when we focus on the shared faults of humanity are those exemplary actions we witness, and the exemplary persons in our communities, that offer us a path forward in our own self-development. Exemplary acts of forgiveness, compassion, grace, courage, creativity and justice have the power to draw our aspirations upward; especially when we consider that there is no reason why we would be incapable of these actions ourselves. But this cannot happen unless we are in the habit of looking for, and taking notice of, moral exemplars in the world around us.

3. Exercising our Moral Imaginations: It can be hard to notice our ethical obligations, or their importance, because we have difficulty imagining how what we do might affect others. In some sense we all know that our personal and professional choices almost always have consequences for the lives of others, whether good or bad. But rarely do we try to really imagine *what it will be like* to suffer the pain that our action is likely going to cause someone – or what it will be like to experience the joy, or relief of pain or worry that another choice of ours might bring. This becomes even harder as we consider stakeholders who live outside of our personal circles and beyond our daily view. The pain of your best friend who you have betrayed is easy to see, and not difficult to imagine before you act - but it is easy *not* to see, and not to imagine, the pain of a person on another continent, unknown to you, whose life has been ruined by identity theft because you knowingly allowed a product with gaping security holes to be released without providing customers a patch. The suffering of that person, and your responsibility for it, would be no less great simply because *you* had difficulty imagining it. Fortunately, our powers of imagination can be increased. Seeking out news, books, films and other sources of stories

about the human condition can help us to better envision the lives of others, even those in very different circumstances from our own. This capacity for *imaginative empathy*, when habitually exercised, enlarges our ability to envision the likely impact of our actions on other stakeholders. Over time, this can help us to fulfill our ethical obligations and to live as better people.

4. Acknowledging Your Own Moral Strength: For the most part, living well in the ethical sense makes life *easier*, not harder. Acting like a person of courage, compassion and integrity is, in most circumstances, also the sort of action that garners respect, trust and friendship in both private and professional circles, and these are actions that we ourselves can enjoy and look back upon with satisfaction rather than guilt, disappointment or shame. But it is inevitable that sometimes the thing that is right will not be the easy thing, at least not in the short term. And all too often our moral will to live well gives out at *exactly* this point – under pressure, we take the easy (and wrong) way out, and try as best we can to put our moral failure and the harm we may have done or allowed out of our minds.

One of the most common reasons why we fail to act as we know we should is that we think we are too *weak* to do so, that we lack the strength to make difficult choices and face the consequences of doing what is right. But this is often more of a self-justifying and self-fulfilling fantasy than a reality; just as a healthy person may tell herself that she simply *can't* run five miles, thus sparing her the effort of trying what millions of others just like her have accomplished, a person may tell herself that she simply *can't* tell the truth when it will greatly inconvenience or embarrass her, or that she simply *can't* help someone in need when it will cost her something she wants for herself. But of course people do these things every day; they tell the morally important truth and take the heat, they sell their boat so that their disabled friend's family does not become homeless, they report frauds from which they might otherwise have benefited financially. These people are not a different species from the rest of us; they just have not forgotten or discounted their own moral strength. And in turn, they live very nearly as they should, *and as we at any time can, if we simply have the will.*

5. Seeking the Company of Other Moral Persons – many have noted the importance of friendship in moral development; in the 4th century B.C. the Greek philosopher Aristotle argued that a virtuous friend can be a 'second self,' one who represents the very qualities of character that we value and aspire to preserve in ourselves.[10] He notes also that living well in the ethical sense requires ethical *actions*, and that activity is generally easier and more pleasurable in the company of others. Thus seeking the company of other moral persons can keep us from feeling isolated and alone in our moral commitments; friends of moral character can increase our pleasure and self-esteem when we do well alongside them, they can call us out when we act inconsistently with our own professed ideals and values, they can help us reason through difficult moral choices, and they can take on the inevitable challenges of ethical life with us, allowing us to weather them together. Aside from this, and as compared with persons who are ethically compromised, persons of moral character are direct sources of pleasure and comfort – we benefit daily

[10] Aristotle, *Nicomachean Ethics,* Book VIII.

from their kindness, honesty, mercy, wisdom and courage, just as they find comfort and happiness in ours.

Question 6:1: Of these five moral habits and practices, which do you think you are best at presently? Which of these habits, if any, would you like to do more to cultivate?

Type answer here

Question 6.2: In what specific ways, small or large, do you think adopting some or all of these habits could change a person's personal and professional life?

Type answer here

PART SEVEN

Question 7:1: What might be the end goal of an *ethical* life as a software engineer? What professional goals or other valuable ends could a software engineer achieve by living well in the ethical sense? What personal goals or values could it help such a person achieve? Answer in as much detail as you are able.

Type answer here

PART EIGHT

What are the professional codes of software engineering ethics? How do they actually help us to be ethical in our working lives?

Each professional society of engineers adopts and enforces its own codes of practice, including codes for *ethical* practice; in Appendix A and B below we have included the codes of ethics adopted by the professional societies most relevant to software engineers in the United States: the NSPE (National Society of Professional Engineers) and the ACM/IEEE-CS (Association for Computing Machinery/Institute of Electrical and Electronic Engineers Computing Society).

These codes are *not* meant to serve as formal checklists or exhaustive accounts of how to be an ethical engineer in any given situation; the latter can only be determined through the engineer's skillful, sincere and habitual practice of ethical reflection, analysis and deliberation in his or her professional life. Ethical codes are just one tool that help us to "develop an 'eye' for what would be appropriate" in various circumstances.[11] As another scholar puts it, "The principles of the Code do not constitute an algorithmic Turing machine that solves ethical problems. Professional judgments are still necessary."[12] Good judgment, what Aristotle called *phronesis* or 'practical wisdom', is something that is acquired through a combination of experience, good habits, and conscious attention to ethical concerns. Ethical rules and codes are no substitute for it, nor are they meant to be. In fact, such codes can only be *used effectively* by persons with good judgment. The codes aim simply to express as fully as possible the *scope* of professional actions governed by ethics and to indicate the specific ethical *duties* that engineers of the highest professional standing expect their present and future colleagues to respect, and to fulfill.

In describing the character of exemplary software engineers, one scholar identifies seven qualities of 'superprofessionals' who embody the highest ideals of their field: A *strong sense of individual responsibility, acute awareness* of the world around them, *brutal honesty, resilience under pressure, a heightened sense of fairness, attention to detail while maintaining perspective,* and *pragmatism* in applying professional standards.[13] Each of these qualities contributes to the practical wisdom that allows us to apply ethical codes intelligently and successfully. Yet "Superprofessionals behave ethically not because it's prescribed by a code of conduct, but because not doing so would violate their personal professional standards."[14]

---

[11] Ruth Chadwick, quoted in Rashid, Weckert and Lucas (2009), 39.
[12] Gotterbarn and Miller (2009), 68.
[13] Erdogmus (2009).
[14] Ibid., 6.

Question 8:1: Read over Appendix A and B. Identify two code items from Appendix A and two from Appendix B about which you have either: a question concerning its meaning, a comment about its importance, or a concern about its ability to be well implemented. Pose those questions/comments/concerns below.

Type your four questions/comments/concerns about Appendix A and B here.

Question 8:2: How might the qualities of a 'superprofessional' engineer with practical wisdom, as described in this section, assist such a person in applying these codes successfully? Be specific.

Type your answer here.

APPENDIX A.

NATIONAL SOCIETY OF PROFESSIONAL ENGINEERS

CODE OF ETHICS FOR ENGINEERS

Preamble

Engineering is an important and learned profession. As members of this profession, engineers are expected to exhibit the highest standards of honesty and integrity. Engineering has a direct and vital impact on the quality of life for all people. Accordingly, the services provided by engineers require honesty, impartiality, fairness, and equity, and must be dedicated to the protection of the public health, safety, and welfare. Engineers must perform under a standard of professional behavior that requires adherence to the highest principles of ethical conduct.

I. Fundamental Canons

Engineers, in the fulfillment of their professional duties, shall:

Hold paramount the safety, health, and welfare of the public.

Perform services only in areas of their competence.

Issue public statements only in an objective and truthful manner.

Act for each employer or client as faithful agents or trustees.

Avoid deceptive acts.

Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.

II. Rules of Practice

Engineers shall hold paramount the safety, health, and welfare of the public.

If engineers' judgment is overruled under circumstances that endanger life or property, they shall notify their employer or client and such other authority as may be appropriate.

Engineers shall approve only those engineering documents that are in conformity with applicable standards.

Engineers shall not reveal facts, data, or information without the prior consent of the client or employer except as authorized or required by law or this Code.

Engineers shall not permit the use of their name or associate in business ventures with any person or firm that they believe is engaged in fraudulent or dishonest enterprise.

Engineers shall not aid or abet the unlawful practice of engineering by a person or firm.

Engineers having knowledge of any alleged violation of this Code shall report thereon to appropriate professional bodies and, when relevant, also to public authorities, and cooperate with the proper authorities in furnishing such information or assistance as may be required.

Engineers shall perform services only in the areas of their competence.

Engineers shall undertake assignments only when qualified by education or experience in the specific technical fields involved.

Engineers shall not affix their signatures to any plans or documents dealing with subject matter in which they lack competence, nor to any plan or document not prepared under their direction and control.

Engineers may accept assignments and assume responsibility for coordination of an entire project and sign and seal the engineering documents for the entire project, provided that each technical segment is signed and sealed only by the qualified engineers who prepared the segment.

Engineers shall issue public statements only in an objective and truthful manner.

Engineers shall be objective and truthful in professional reports, statements, or testimony. They shall include all relevant and pertinent information in such reports, statements, or testimony, which should bear the date indicating when it was current.

Engineers may express publicly technical opinions that are founded upon knowledge of the facts and competence in the subject matter.

Engineers shall issue no statements, criticisms, or arguments on technical matters that are inspired or paid for by interested parties, unless they have prefaced their comments by explicitly identifying the interested parties on whose behalf they are speaking, and by revealing the existence of any interest the engineers may have in the matters.

Engineers shall act for each employer or client as faithful agents or trustees.

Engineers shall disclose all known or potential conflicts of interest that could influence or appear to influence their judgment or the quality of their services.

Engineers shall not accept compensation, financial or otherwise, from more than one party for services on the same project, or for services pertaining to the same project, unless the circumstances are fully disclosed and agreed to by all interested parties.

Engineers shall not solicit or accept financial or other valuable consideration, directly or indirectly, from outside agents in connection with the work for which they are responsible.

Engineers in public service as members, advisors, or employees of a governmental or quasi-governmental body or department shall not participate in decisions with respect to services solicited or provided by them or their organizations in private or public engineering practice.

Engineers shall not solicit or accept a contract from a governmental body on which a principal or officer of their organization serves as a member.

Engineers shall avoid deceptive acts.

Engineers shall not falsify their qualifications or permit misrepresentation of their or their associates' qualifications. They shall not misrepresent or exaggerate their responsibility in or for the subject matter of prior assignments. Brochures or other presentations incident to the solicitation of employment shall not misrepresent pertinent facts concerning employers, employees, associates, joint venturers, or past accomplishments.

Engineers shall not offer, give, solicit, or receive, either directly or indirectly, any contribution to influence the award of a contract by public authority, or which may be reasonably construed by the public as having the effect or intent of influencing the awarding of a contract. They shall not offer any gift or other valuable consideration in order to secure work. They shall not pay a commission, percentage, or brokerage fee in order to secure work, except to a bona fide employee or bona fide established commercial or marketing agencies retained by them.

III. Professional Obligations

Engineers shall be guided in all their relations by the highest standards of honesty and integrity.

Engineers shall acknowledge their errors and shall not distort or alter the facts.

Engineers shall advise their clients or employers when they believe a project will not be successful.

Engineers shall not accept outside employment to the detriment of their regular work or interest. Before accepting any outside engineering employment, they will notify their employers.

Engineers shall not attempt to attract an engineer from another employer by false or misleading pretenses.

Engineers shall not promote their own interest at the expense of the dignity and integrity of the profession.

Engineers shall at all times strive to serve the public interest.

Engineers are encouraged to participate in civic affairs; career guidance for youths; and work for the advancement of the safety, health, and well-being of their community.

Engineers shall not complete, sign, or seal plans and/or specifications that are not in conformity with applicable engineering standards. If the client or employer insists on such unprofessional conduct, they shall notify the proper authorities and withdraw from further service on the project.

Engineers are encouraged to extend public knowledge and appreciation of engineering and its achievements.

Engineers are encouraged to adhere to the principles of sustainable development1 in order to protect the environment for future generations.

Engineers shall avoid all conduct or practice that deceives the public.

Engineers shall avoid the use of statements containing a material misrepresentation of fact or omitting a material fact.

Consistent with the foregoing, engineers may advertise for recruitment of personnel.

Consistent with the foregoing, engineers may prepare articles for the lay or technical press, but such articles shall not imply credit to the author for work performed by others.

Engineers shall not disclose, without consent, confidential information concerning the business affairs or technical processes of any present or former client or employer, or public body on which they serve.

Engineers shall not, without the consent of all interested parties, promote or arrange for new employment or practice in connection with a specific project for which the engineer has gained particular and specialized knowledge.

Engineers shall not, without the consent of all interested parties, participate in or represent an adversary interest in connection with a specific project or proceeding in which the engineer has gained particular specialized knowledge on behalf of a former client or employer.

Engineers shall not be influenced in their professional duties by conflicting interests.

Engineers shall not accept financial or other considerations, including free engineering designs, from material or equipment suppliers for specifying their product.

Engineers shall not accept commissions or allowances, directly or indirectly, from contractors or other parties dealing with clients or employers of the engineer in connection with work for which the engineer is responsible.

Engineers shall not attempt to obtain employment or advancement or professional engagements by untruthfully criticizing other engineers, or by other improper or questionable methods.

Engineers shall not request, propose, or accept a commission on a contingent basis under circumstances in which their judgment may be compromised.

Engineers in salaried positions shall accept part-time engineering work only to the extent consistent with policies of the employer and in accordance with ethical considerations.

Engineers shall not, without consent, use equipment, supplies, laboratory, or office facilities of an employer to carry on outside private practice.

Engineers shall not attempt to injure, maliciously or falsely, directly or indirectly, the professional reputation, prospects, practice, or employment of other engineers. Engineers who believe others are guilty of unethical or illegal practice shall present such information to the proper authority for action.

Engineers in private practice shall not review the work of another engineer for the same client, except with the knowledge of such engineer, or unless the connection of such engineer with the work has been terminated.

Engineers in governmental, industrial, or educational employ are entitled to review and evaluate the work of other engineers when so required by their employment duties.

Engineers in sales or industrial employ are entitled to make engineering comparisons of represented products with products of other suppliers.

Engineers shall accept personal responsibility for their professional activities, provided, however, that engineers may seek indemnification for services arising out of their practice for other than gross negligence, where the engineer's interests cannot otherwise be protected.

Engineers shall conform with state registration laws in the practice of engineering.

Engineers shall not use association with a nonengineer, a corporation, or partnership as a "cloak" for unethical acts.

Engineers shall give credit for engineering work to those to whom credit is due, and will recognize the proprietary interests of others.

Engineers shall, whenever possible, name the person or persons who may be individually responsible for designs, inventions, writings, or other accomplishments.

Engineers using designs supplied by a client recognize that the designs remain the property of the client and may not be duplicated by the engineer for others without express permission.

Engineers, before undertaking work for others in connection with which the engineer may make improvements, plans, designs, inventions, or other records that may

justify copyrights or patents, should enter into a positive agreement regarding ownership.

Engineers' designs, data, records, and notes referring exclusively to an employer's work are the employer's property. The employer should indemnify the engineer for use of the information for any purpose other than the original purpose.

Engineers shall continue their professional development throughout their careers and should keep current in their specialty fields by engaging in professional practice, participating in continuing education courses, reading in the technical literature, and attending professional meetings and seminars.

*Footnote 1 "Sustainable development" is the challenge of meeting human needs for natural resources, industrial products, energy, food, transportation, shelter, and effective waste management while conserving and protecting environmental quality and the natural resource base essential for future development.

APPENDIX B.

SOFTWARE ENGINEERING CODE OF ETHICS AND PROFESSIONAL PRACTICE (Version 5.2)

ADOPTED BY THE ACM/IEEE-CS JOINT TASK FORCE ON SOFTWARE ENGINEERING ETHICS AND PROFESSIONAL PRACTICES

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 policy makers, 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.

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.

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.

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.

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 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.

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.

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.

This Code was developed by the ACM/IEEE-CS joint task force on Software Engineering Ethics and Professional Practices (SEEPP):

Executive Committee: Donald Gotterbarn (Chair), Keith Miller and Simon Rogerson;

Members: Steve Barber, Peter Barnes, Ilene Burnstein, Michael Davis, Amr El-Kadi, N. Ben Fairweather, Milton Fulghum, N. Jayaram, Tom Jewett, Mark Kanko, Ernie Kallman, Duncan Langford, Joyce Currie Little, Ed Mechler, Manuel J. Norman, Douglas Phillips, Peter Ron Prinzivalli, Patrick Sullivan, John Weckert, Vivian Weil, S. Weisband and Laurie Honour Werth.

APPENDIX C.

BIBLIOGRAPHY/FURTHER READING

On Software Engineering/Computer Ethics

Online Resources

ABET (Accreditation Board for Engineering and Technology). http://www.abet.org/

ACM/IEEE-Computer Society. *Software Engineering Code of Ethics and Professional Practice*. Version 5.2. http://www.acm.org/about/se-code

National Academy of Engineering's Center for Engineering, Ethics and Society (CEES). http://www.nae.edu/26187.aspx

NSPE (National Society of Professional Engineers). *Engineering Ethics*. http://www.nspe.org/Ethics/index.html

Online Ethics Center for Engineering and Research. http://www.onlineethics.org/

Books and Edited Collections (in reverse chronological order)

Spinello, Richard (2014) Cyberethics: Morality and Law in Cyberspace, 5th ed., Jones & Bartlett; 246 pages.

Tavani, Herman T. (2013) Ethics and Technology: Controversies, Questions, and Strategies in Ethical Computing, 4th Ed., John Wiley & Sons; 454 pages.

Floridi, Luciano, ed. (2010) The Cambridge Handbook of Information and Computer Ethics, Cambridge University Press; 342 pages.

Johnson, Deborah G., ed. (2009) Computer Ethics, 4th ed., Pearson; 216 pages.

Himma, Kenneth E. and Tavani, Herman T., eds., (2008) The Handbook of Information and Computer Ethics, John Wiley & Sons; 702 pages.

Weckert, John, ed. (2007) Computer Ethics, Ashgate; 516 pages.

Spinello, Richard and Tavani, Herman T. eds. (2004) Readings in Cyberethics, Jones and Bartlett; 697 pages.

Bynum, Terrell Ward and Rogerson, Simon, eds. (2004) Computer Ethics and Professional Responsibility, Blackwell; 378 pages.

Johnson, Deborah G. and Nissenbaum, Helen, eds. (1995) Computers, Ethics & Social Values, Prentice Hall; 656 pages.

Articles and Encyclopedia Entries (in reverse chronological order)

Grodzinsky, Frances S., Miller, Keith W. and Wolf, Marty J. (2012) "Moral responsibility for computing artifacts: "the rules" and issues of trust." *ACM SIGCAS Computers and Society*, 42:2, 15-25.

Bynum, Terrell (2011) "Computer and Information Ethics", *The Stanford Encyclopedia of Philosophy*, Edward N. Zalta (ed.), http://plato.stanford.edu/archives/spr2011/entries/ethics-computer/

Berenbach, Brian and Broy, Manfred (2009). "Professional and Ethical Dilemmas in Software Engineering." *IEEE Computer* 42:1, 74-80.

Erdogmus, Hakan (2009). "The Seven Traits of Superprofessionals." *IEEE Software* 26:4, 4-6.

Hall, Duncan (2009). "The Ethical Software Engineer." *IEEE Software* 26:4, 9-10.

Rashid, Awais, Weckert, John and Lucas, Richard (2009). "Software Engineering Ethics in a Digital World." *IEEE Computer* 42:6, p. 34-41.

Wolf, Marty J., Miller, Keith W and Grodzinsky, Frances S. (2009) "Free, source-code-available or proprietary: an ethically charged, context-sensitive choice." *ACM SIGCAS Computers and Society*, 39:1, 15-26.

Gotterbarn, Donald and Miller, Keith W. (2009) "The public is the priority: making decisions using the Software Engineering Code of Ethics." *IEEE Computer*, 42:6, 66-73.

Gotterbarn, Donald. (2008) "Once more unto the breach: Professional responsibility and computer ethics." *Science and Engineering Ethics* 14:1, 235-239.

Johnson, Deborah G. and Miller, Keith W. (2004) "Ethical issues for computer scientists." *The Computer Science and Engineering Handbook* 2nd Ed,, A. Tucker, ed. Springer-Verlag, 2.1-2.12.

Gotterbarn, Donald (2002) "Software Engineering Ethics," *Encyclopedia of Software Engineering*, 2nd ed., John Marciniak ed., John Wiley & Sons.

<u>On General Philosophical Ethics</u>

Aristotle (2011). *Nicomachean Ethics.* Translated by R.C. Bartlett and S.D. Collins. Chicago: University of Chicago Press.

Cahn, Steven M. (2010). *Exploring Ethics: An Introductory Anthology*, 2nd Edition. Oxford: Oxford University Press.

Shafer-Landau, Russ (2007). *Ethical Theory: An Anthology.* Oxford: Blackwell Publishing.

An Introduction to Software Engineering Ethics

MODULE AUTHORS:
Shannon Vallor, Ph.D. Associate Professor of Philosophy, Santa Clara University
SPECIAL CONTRIBUTOR TO INTRODUCTION:
Arvind Narayanan, Ph.D. Assistant Professor of Computer Science, Princeton University

CLASSROOM DISCUSSION AND CASE STUDY EXERCISES

EXERCISE 1: RUSH JOBS AND RED LIES: CASCADING ETHICAL FAILURES

In their article "Professional and Ethical Dilemmas in Software Engineering," Brian Berenbach and Manfred Broy discuss various ways in which software engineers can find themselves pressured to compromise professional and ethical standards. They include cases of "Mission Impossible" (being asked to create or accept a product schedule that is clearly impossible to meet), "Mea Culpa" (delivering products without key functionality or with known defects), "Rush Jobs" (delivering products of subpar quality to meet schedule pressures), "Red Lies" (telling clients or management known falsehoods about product schedule or performance), "Fictionware" (promising features that are infeasible), and "Nondiligence" (inadequate review of requests for proposals, contracts or specifications).[15]

Consider the following scenario, adapted from several real-life cases:

> A mid-level sales manager at LifeDesign, a medical systems engineering firm, receives a request for proposal (RFP) to develop a radiation-delivery system for use in outpatient hospital settings. The RFP specifies that the product must have a guaranteed failsafe mechanism to prevent radiation overdoses due to user input error. The sales manager reviews the RFP with his development team, who tell him that A) the system cannot be built to specifications within the desired timeframe and B) there is no way in this kind of system to build in a *guaranteed* failsafe against operator error; proper radiation dosages vary so much by patient that there is no way for the system to enforce safe limits by itself; it *must* rely on accurate user input of the dosage instructions on the prescription. The best we can offer, the team tells him, is a warning system that will loudly prompt the user to verify and double-verify the correct input. But, the team tells him, it is well-known that such warnings are often mindlessly canceled by users who find them annoying, so this mechanism is hardly guaranteed to prevent operator error. The sales manager goes back to the client with a proposal that promises delivery of all functionality within the specified time frame.
>
> When they receive the proposal, which has been accepted and contracted with the client, the engineering team begins to design the system. But it is three

[15] Brian Berenbach and Manfred Broy (2009), "Professional and Ethical Dilemmas in Software Engineering," *Computer*, 42:1, 74-80.

weeks into the work before anyone notices the delivery date – the design team leader immediately appeals to the sales manager, reminding him that he was told this was an impossible date to meet. He tells her that this is unfortunate but that they are under contract now, and there will be severe financial penalties for a late delivery. He reminds her that upper management won't be happy with any of them in that case, and tells her that her team had better find a way to get it done. The team leader goes back to her group, frustrated but resigned, and tells them to get it done somehow. No one has even noticed yet that the contract also includes language about the guaranteed failsafe against bad user input.

Halfway through the delivery schedule, the sales manager and design team leader sit down with the client for an update. Despite being well behind schedule, they put on a good show for the client, who leaves confident that all is as it should be and that the product will deliver full functionality on time. Weeks before the due date, the team leader is telling her team that their jobs depend on an on-time delivery. Her engineering team has resorted to desperate measures – especially the group responsible for the system software, who are taking shortcuts, hiding system errors, doing sloppy coding and subcontracting work out to third-parties without proper review of their qualifications. When the product is finally assembled, there is no time to test the full system in the actual client/user environment, so the quality-assurance team of LifeDesign relies on computer simulations to test its operation. These simulations make many flawed assumptions due to the rush schedule and a lack of information about the user context/environment. None of the simulations assume erroneous dosage input by users.

The product is delivered on time, and represented as having full functionality. As a result, the client tells its therapists using the new system that it has an advanced failsafe mechanism that will prevent radiation overdoses from user error. Two years into its use, dozens of patients receiving radiation from the system, including young children and others with curable diseases, have been poisoned by radiation overdoses. Those who are not already ill or dying from overdose-related effects must now live with the knowledge that they have a severely elevated chance of developing fatal bladder or kidney cancers from the overdoses. An external review of the system by federal regulators and litigators reveals that not only was there no 'guaranteed' failsafe mechanism, the warning system that did exist was so buggy that it would often not be triggered at all, or would behave so erratically that most users dismissed it whenever it was triggered. Lawsuits are piling up, the media is on the story, and everyone is pointing fingers – at the radiation therapists, at the client, and *especially* at LifeDesign.

Question 1:1: Which of the various pressures identified by Berenbach and Broy at the start of this section contributed to the outcome of this case?

Type answers here

Question 1:2: Who were the various *stakeholders* whose interests were ethically significant here? What *were* the interests of each stakeholder? Whose interests should have taken precedence in the minds of the relevant actors, and why?

Type answers here

Question 1:3: Who at LifeDesign had the power to prevent this outcome? The sales manager? The design team manager? The manager of the software group? The individual members of that group? The quality-assurance team? The CEO? Go through the list and discuss what each person, at each level, could have done to effectively prevent this outcome.

Type answers here

Question 1:4: Put yourselves in the shoes of the software engineers employed on this project. Discuss together how the outcome of this case would affect you, personally and professionally. How would you feel about your friends, family, neighbors and mentors learning that you were involved in the scandal, and that your work was implicated in the suffering and deaths of many innocent people? What would you say to them to explain yourself? Would any explanation be adequate?

Type answers here

EXERCISE 2: PRIVACY BY DESIGN (OR NOT): GOOGLE STREET VIEW

Along with intellectual property/copyright concerns, privacy is one of the most commonly discussed issues of ethical and legal concern with software. There are many definitions of what constitutes privacy: they include control over one's personal information; the 'right to be forgotten,' to be left alone or to have a measure of obscurity; the integrity of the context in which your personal information is used; and the ability to form your own identity on your own terms. Each of these, along with many other potential definitions, captures something important about privacy. There is also increasing debate about the extent to which new technologies are changing our expectations of privacy, or even how much we value it.

Regardless, privacy is in many contexts a legally protected right, and, in all contexts, among those interests that stakeholders may legitimately expect to be acknowledged and respected. The pressures that Web 2.0, 'Big Data,' cloud computing and other technological advances are putting on privacy will continue to make headlines for the foreseeable future, and software engineers will continue to struggle to balance the legitimate desire for expanding software functionality with the ethical requirements of privacy protection. This is complicated by the spread of 'dual-use' technologies with open-ended and adaptable functionalities, and technologies that offer a scaffold upon which third-party apps can be built.

Among the most famous cases to reveal these challenges is Google Street View:

> In 2007, Google launched its StreetView feature, which displays searchable aerial and street-view photographs of neighborhoods, city blocks, stores, and even individual residences. From the very beginning privacy concerns with Google's technology were evident; it did not take long for people to realize that the feature displayed photographs of unwitting customers leaving adult bookstores and patients leaving abortion clinics, children playing naked, adults sunning themselves topless in their backyards, and employees playing hooky from work. Moreover, it was recognized that the display of these photos was being used by burglars and other criminals to identify ideal targets. Although Google did initially think to remove photos of some sensitive locations, such as domestic violence shelters, it was initially very difficult for users to request removal of photos that compromised their privacy. After an initial outpouring of complaints and media stories on its privacy problems, Google streamlined the user process for requesting image removal. This still presupposed, however, that users were *aware* of the breach of their privacy.
>
> In 2010, StreetView became the center of a new privacy scandal; it was discovered that software used in Google vehicles doing drive-by photography had been collecting personal data from unencrypted Wi-Fi networks, including SSID's, device identifiers, medical and financial records, passwords and email content. Initially Google claimed that this data had not been collected, later they said that only 'fragments' of such data had been retained, yet eventually they conceded that complete sets of data had not only been collected but stored. At one point Google blamed the breaches on a single 'rogue engineer,' though

later it was learned that he had communicated with his superiors about the Wi-Fi data collection. As of 2012, 12 countries had launched formal investigations of Google for breaches of privacy or wiretap laws, at least 9 have determined that its laws were violated.[16] In the U.S., Google was fined $25,000 for obstructing its investigation into the matter. More recently, Google settled a lawsuit brought by 38 states over the breaches for $7 million (a tiny fraction of its profits). As part of the settlement, Google acknowledged its culpability in privacy breaches, and promised to set up an annual "privacy week" for its employees along with other forms of privacy training and education.[17]

Question 2:1: What forms of harm did members of the public suffer as a result of Google StreetView images? What forms of harm could they have suffered as a result of Google's data-collection efforts?

Type answers here

[16] Electronic Privacy Information Center, "Investigations of Google StreetView," http://epic.org/privacy/streetview/. Accessed March 18, 2013.
[17] David Streitfeld, "Google Concedes that Drive-By Prying Violated Privacy," *New York Times*, March 12, 2013.

Question 2:2: What institutional *and* professional choices might have been responsible for Google StreetView's violations of public privacy?

Type answers here

Question 2:3: What ethical strategies from your earlier reading in this unit could Google engineers and managers have used to produce a more optimal balance of functionality and ethical design? Could one or more Google 'superprofessionals' have prevented the privacy breaches, and if so, how?

Type answers here

Question 2:4: What can Google do *now* to prevent similar privacy issues with its products in the future?

Type answers here

EXERCISE 3: ANTI-PATTERNS, PROFESSIONALISM AND ETHICS

'Anti-patterns' are engineering or business habits, techniques and solutions that are generally considered substandard, likely to backfire or generate more problems, unreliable, or otherwise indicative of poor professional conduct. The term, then, represents the opposite of what are known as 'best practices.' Hundreds of anti-patterns have been named. Some of them are exclusive to software engineering, while others are general to business and other social institutions. Here are just some of the common anti-patterns that can affect software engineers:

- **Analysis Paralysis** (a risk-averse pattern of unending analysis/discussion that never moves forward to the actual decision phase)
- **Blind Coding/Blind Faith** (implementing a bug fix or subroutine without ever actually testing it)
- **Boat Anchor** (a totally useless piece of software or hardware that you nevertheless keep in your design, often due to its initial cost)
- **Bystander Apathy** (everyone can see impending disaster, but no one is motivated to do anything about it)
- **Cut and Paste Programming** (largely self-explanatory; reusing/cloning code)
- **Death March** (a project everyone knows is doomed to fail but is ordered to keep working on anyway)
- **Design by Committee** (no intelligent vision guiding and unifying the project)
- **Error Hiding** (overriding the display of error messages with exception handling, so neither the user nor tech support can actually see them)
- **Escalation of Commitment** (adopting a misguided goal or poor strategy, then completely refusing to rethink it despite clear evidence of its failure)
- **Improbability Factor** (refusing to expend resources on a fix on the assumption that the problem is unlikely to actually occur in use)
- **Input Kludge** (failure of a software program to anticipate and handle invalid or incorrect user input)
- **Gold Plating** ('gilding the lily'; continuing to develop a product or adding features that offer insufficient value to justify the effort)
- **Golden Hammer** (using the same favored solution at every opportunity, whether it's the appropriate tool for the job or not)
- **Lava Flow** (old code, often undocumented and with its function poorly understood, therefore left in)
- **Moral Hazard** (insulating decision makers from the risks/negative consequences of their decisions)
- **Mushroom Management** (keeping non-management employees in the dark, information-starved)
- **Software Bloat** (successive iterations of software using more and more memory/power/other resources with little or no added functionality)
- **Spaghetti Code/Big Ball of Mud** (unstructured, messy code not easily modified or extended)

Question 3:1: What makes the above anti-patterns not *just* potential sources of poor professional practice, but potential sources of *unethical* conduct as well? In general, how do these anti-patterns relate to the NSPE and ACM/IEEE-CS codes, and how do they show the connection between professional and ethical conduct?

Type answers here

Question 3:2: Which of these anti-patterns are most easy to see as causes of unethical engineering conduct? Which are more difficult to connect to lapses of ethics? Are there any that you think aren't connected to ethics at all? Discuss this among your group.

Type answers here

Question 3:3: Which of these anti-patterns would you think are among the most difficult to avoid in software engineering practice, and why?

Type answers here

Question 3.4: Construct your *own* fictional case study using these anti-patterns. Create an engineering scenario in which at least *three* of these anti-patterns interact and lead to an outcome that is profoundly unethical. Describe the harm that results, and then identify at least two interventions or measures that members of the organization could nd *hould* have taken to prevent that outcome.

Type answers here