Software Engineering Ethics

Chapter · January 2002

DOI: 10.1002/0471028959.sof314

1 author:

Donald Gotterbarn
East Tennessee State University
179 PUBLICATIONS **1,627** CITATIONS

SEE PROFILE

SOFTWARE ENGINEERING ETHICS

Donald Gotterbarn
Software Engineering Ethics Research Institute
Article on Software Engineering Ethics for the *Encyclopedia of Software Engineering*

OVERVIEW

Software engineering ethics can be approached from three directions. First, it can describe the activity of software engineers making practical choices that affect other people in significant ways. Second, it can be used to describe a collection of principles, guidelines, or ethical imperatives that guide or legislative action, and third, it can be used to name a discipline that studies the relationship between the other two senses of ethics. Software engineering ethics is clearly both an activity and a body of principles. The discipline of software engineering ethics that studies this activity and formalizes these principles, however, is in its infancy.

Software Engineering Ethical Activity

To avoid confusion, “ethics”, as understood here, addresses any intentional action that impacts negatively or positively the lives and values of others. Software engineering conceives of itself primarily as a technical discipline that develops software. There are a variety of names, such as information systems analyst, for those who engage in professional software development. Regardless of the title used, the focus of software engineering activity is primarily on the technical adequacy of the products developed. But the fact that roughly one billion people depend on software systems to effectively conduct their daily lives (Reed, 2000) has led many in computing to give more attention to the non-technical aspects and to wrestle with the ethical impact of their daily decisions and the values imbedded therein. The relationship between computers and ethics can be described as occurring when humans make decisions about computers, and those decisions affect people's lives. Human values are linked to technical decisions in this way. The activity of Professional Software Engineering Ethics takes place when any decision made by computing professionals during the design, development, construction and maintenance of computing artifacts affect other people. These decisions may be made by individuals, teams, management, or the profession. The software engineering decision how to design the release of an airbag will affect the lives of others. This technical decision is also guided by an ethical decision about human values. Barry Boehm (1981) begins his work on software engineering economics with an anecdote about how the failure to consider human values affected the development of software for a high school attendance system. His work on this project led him to see that software engineers have "… an opportunity to make a significant positive impact on society, simply by becoming more sensitive to the long-term human relations implications of our work and incorporating this sensitivity into our software designs and products." (Boehm, 1981)

Software Engineering Ethical Principles

The ethical activity involved in technical decisions should be based on an understanding of the impact of those decisions. Often decisions are made while only looking at the technical issues, but those decisions impact others in negative and positive ways. A good software engineering decision requires an awareness of and a careful consideration of

both the technical and the ethical dimensions The technical decisions should be consciously guided by values. An absence of the awareness of this relationship between technology and values means that the values related to the technology are haphazard. Professions describe these values in Codes of Ethics, Codes of Conduct and Codes of Practice. These codes serve many functions related to a profession including providing guidance for practitioner's ethical choices and educating practitioners and the public about the ethical standard of the profession.

Codes of ethics are statements about the interaction of technology and values. They are also intended to insure that this interaction is not haphazard. Some codes, such as the Australian Computer Society Code of Ethics carries legal weight to insure structured interaction, while other codes, such as the Software Engineering Code of Ethics and Professional Practice, rely on normative pressures. The goal of all of these codes is to make sure that the activity of practicing professionals as individual, teams, management, or professional organizations, advance and protect human values rather than do damage to them.

The Discipline of Software Engineering Ethics

The discipline of software engineering ethics studies the interaction of the ethical principles and technology to examine the ways in this which this interaction affects society, its citizens, and software engineering products. As a discipline, software engineering ethics illustrates, refines and further explores the relationship between these principles and the developing technical practice.

Development of Software Engineering Ethics

Software engineering is an emerging profession, which is self-conscious of its impact on society and its resulting obligations. The fact that computing has a significant daily impact on the lives of billions of people has led many in computing to wrestle with their ethical obligations. In the 1990s many organizations have revisited their codes of conduct or codes of ethics and revised them in a way that more clearly states an ethical obligation to the public at large. For example, the Association for Computing Machinery (ACM), the Australian Computer Society, the British Computer Society (BCS) , the IEEE Computer Society (IEEE-CS) , and the New Zealand Computer Society have all refined their Codes of Ethics. A broad based effort to professionalize software engineering was started by the IEEE Computer Society, and later joined by the ACM (Gotterbarn a, 1996). The initial goals of this organization were to:

1. Adopt Standard Definitions
2. Define Required Body of Knowledge and Recommended Practices (in electrical engineering, for example, electromagnetic theory is part of the body of knowledge while the National Electrical Safety Code is a recommended practice.)
3. Define Ethical Standards
4. Define Educational Curricula for (a) undergraduate, (b) graduate (MS), and (c) continuing education (for retraining and migration). (Gotterbarn b, 1996)

These goals are consistent with the goals of other professions. One of the key elements in any profession is the recognition of its ethical and moral responsibilities to its clients, to society, and to the profession itself. Many professions, such as engineering and medicine, declare these responsibilities publicly in a code of ethics and then later require training in professional ethics in order to enter the profession. For example, the IEEE-CS Certification as a Software Engineering Professional requires adherence to the Software Engineering Code of Ethics and Professional Practice.

However, this interest in ethics was not always a major element of software engineering. Software engineering had defined itself as "the establishment and use of sound engineering principles [methods] in order to obtain economically software that is reliable and works on real machines."(Bauer, 1972) does not explicitly point to the existence of ethical issues. The IEEE Standard 610.12 definition also has the same lacuna.

“Software Engineering:

(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

(2) The study of approaches as in (1).“

As the discipline matured toward a profession and the impact of its product enlarged, the descriptions of the discipline begin explicitly to mention places where ethical issues might arise. When software engineering is described as a discipline which "requires understanding and application of engineering principles, design skills, good management practice, computer science, and mathematical formalism.(BCS, 1989)", some areas of ethical concern such as good management practices begin to emerge. These early descriptions do not give prominence to ethical issues. Bauer's definition does not mention satisfaction of the clients needs or satisfying the standards of the profession, yet these are implied in terms like "reliable", "economical" and "works". The BCS’s complete definition does include some reference to the software engineer's responsibility to the profession. As the impact of software engineering has broadened so has the recognition of the responsibilities of a software engineer.

These responsibilities formulated in the Software Engineering Code of Ethics and Professional Conduct (SECODE 1999) were summarized as:

> Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:
>
> 1 PUBLIC - Software engineers shall act consistently with the public interest.

2 CLIENT AND EMPLOYER - Software engineers shall act in a manner that is in the best interests of their client or employer and that is consistent with the public interest.

3 PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

4 JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.

5 MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

6 PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

7 COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.

8 SELF - Software engineers shall participate in lifelong learning regarding the practice of their profession and promote an ethical approach to the practice of the profession.

This set of principles, developed and reviewed by software engineers from every continent expresses the software engineers commitment to a level of professional care. The professionalization of any discipline involves the realization that being a professional involves more than the rigid application of the most recent technology to the artifacts of that discipline (Ford, 1996). The practice of medicine is more than the application of drugs to the human body. Physicians are concerned with the well-being of their patients. The professional architect does not merely apply principles of structural stress when designing a building but is concerned with the effect of the design on the people who will be using the building. In the professionalization of a discipline there is: 1) a realization of the impact on society of the application of the special skills, 2) knowledge by the practitioner of that profession's standards and, 3) a realization of the responsibility that comes with the privilege of being allowed to apply those skills.

This kind of realization in software engineering is seen in the definition offered by the Software Engineering Institute. It defines software engineering as a "form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems(Ford, 1990)." But software engineering, as a subset of engineering, is the creating and building of "cost-effective solutions to practical problems in the service of mankind"(Ford, 1990). Ford goes on to point out explicitly that cost-effective does not mean making something of marginal quality to minimize time and resources, rather, "cost-effective ...implies getting good value for the resources invested; this value includes quality (Ford, 1990).

SOFTWARE ENGINEERING ETHICS

Software engineering has been described as "The disciplined application of engineering, scientific, and mathematical principles, methods, and tools to the economical production of quality software" .(Humphrey, 1989). As a software engineer applies these principles, human values become intertwined with technical decisions. Software engineering ethics studies the interactions of human values and technical decisions involving computing. Software engineering as an engineering-like discipline has a direct connection to ethics.

Software engineering is an applied science. Every product of software engineering involves people and so at any stage of product development the users of the intermediate and final products has to be kept in mind. Software engineers have obligations to the users of their products, which include not only the implemented system but also includes other products such as requirements, software project management plans, specifications, designs, documentation, test suites, programs, user manuals, and training materials.

In developing these software engineering artifacts, each decision is a compromise that is affected by such constraints as available budget, clients needs, available software, reliability requirements, environmental considerations, societal effects, and even political realities. All of these make the job of the professional software engineer more difficult and requires more subjective judgment. But there are some guidelines for making these judgments.

> 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. (SE Code, 1999)

Software engineering ethics establishes principles of conduct that members of the profession are expected to observe in the practice of software engineering. Software engineering ethics are related to two basic aspects of the software engineer's role. On the one hand, the software engineer is employed by clients to serve their interests; on the other, the engineer is participating in an important social function. He or she represents his or her clients' interests but cannot act in a way that endangers society or is contrary to the principles of good software engineering. Sometimes the interests of the client and

society conflict, and the technical principles of software engineering do not address these conflicts. For example, suppose, a software engineer's work on an internet vote-tabulating program has not been sufficiently tested and it is the day before elections. The choice between 1) releasing an uncertified product and continuing testing while people are using the program to vote and 2) failing to release the product and thereby disenfranchising people is not a purely technical decision. To deliver the system will meet the clients' deadlines but delivering it might not be in the best interests of those who are using the system to vote. There are sets of competing values that must be adjudicated. Sometimes these decisions are resolved based on technical knowledge. In the above case, the quality and coverage of the completed testing is significant.

Software engineering ethics covers a wide ethical spectrum. It participates in general, professional, and technical ethics. His is shown in a taxonomy of software engineering ethics which includes some applied examples. This taxonomy is followed by a brief discussion of future trends in software engineering ethics.

New applications of computer technology to familiar situations generate ethical puzzles. For example, sociologists are puzzling about computer monitoring of employees. The use of computers in this monitoring does not make it an issue in computer ethics. The ethical questions of computer monitoring are the same ethical questions involved in monitoring employees through one-way mirrors. Most of these so-called issues in software engineering ethics and computer ethics have analogies with general ethical issues and are resolved when one finds an appropriate analogous situation (Johnson, 2001). Reasoning in ethics frequently employs analogical reasoning. A major problem for software engineering ethics occurs when it is not clear which analogy to apply. For example, if an ISP is analogous to a publisher then like a publisher it is responsible for editing content. If an ISP is analogous to a telephone switch then it is not responsible for editing content. How does software engineering ethics approach the problem of deciding which type of analogy is most appropriate for ethical decision making? The approach is based on engineering ethics and its commitment to both technical quality and the health safety and welfare of the public.

A MODEL FOR SOFTWARE ENGINEERING ETHICS

A consistent and manageable view of software engineering ethics is derived from a concept of ethics as "a public system applying to all rational persons governing behavior which affects others and which has the minimization of evil as its end..."(Gert, 1998). The behavior of professional software engineers, in so far as it affects others, is guided by a set of principles which are designed to minimize evil and encourage good. This definition ties software engineering ethics to the individual rather than to machines. Furthermore, using some elements of professional ethics as a model has the advantage of bringing software engineering ethics within the horizon of action of the practice of the individual moral software engineer. Software engineering ethics is a type of professional ethics. Software engineering is often a profession practiced by teams in managed situations, so as we shall see a complete concept of software engineering ethics includes ethical activity and principles related to the action of teams and the actions of management. The relation

between software engineering ethics and other professional ethics can be understood by developing a partial taxonomy of ethics.

ETHICS AND PROFESSIONAL ETHICS

General Ethics

In the first category, software engineering ethics participates in a very general sense of ethics, which includes most human interaction. This is the concept of ethics as a means to facilitate the interactions of people living in a society by placing obligations of action and avoidance of action on them. In any given society ethics regulates individual behavior in those circumstances where it cannot be or as yet has not been regulated by law. Ethics depends entirely on voluntary acceptance of established rules and is not enforced by the state. In this category, ethics has human well-being as one of its primary goals. Some principles in this category include hurt no one, be honest, be fair, and keep your promises. These are the moral foundations for our obligation to complete a contract and our obligation to be concerned with the user of our artifacts. It also justifies management's obligation to "5.02. Ensure that software engineers are informed of standards before being held to them." (SE Code 1999)

Professional Ethics and General Ethics

The second category of professional ethics has elements in common with the first category of general ethics. Some occupations, which have a significant impact on human well being, require a high degree of special knowledge to produce a product or service. People in these occupations are called professionals.

Professional ethics is a category of ethics that is required of all professionals, regardless of their particular profession. Architects, doctors, engineers, and lawyers are all bound by professional ethics. This broad sense of professional ethics is derived from the concept of a profession. There are several marks of a professional. They generally include a special skill or knowledge to produce a product or service, a commitment to public service, a commitment to cause no harm, membership in a representative organization, adherence to a code of ethics, being licensed by a representative organization, and autonomous judgement based on this specialized knowledge (Ford, 1996). A profession is permitted by society to lay exclusive claim to a body of knowledge because through the utilization of the knowledge, society as a whole gains. This claim to exclusive knowledge is only justified if indeed society does gain. In exchange for this exclusive claim, a profession assumes an obligation to serve society (Bayles, 1981). These obligations are generally articulated in professional codes of ethics.

The principle of responsibility to one's profession and to one's client does not change across professions. This category of ethics is common to all of the technical professions. In some professional codes the role of the professional is defined in terms of a relationship between consultant and client and the primary concerns are obligations to clients and to

the profession as a whole. This emphasis directs ethical concerns toward issues of group loyalty (do not criticize another engineer), paternalism (the doctor will decide what is best for the patient), and advertising (do not compete with a fellow professional by advertising) (Martin 1989).

There has been a new importance given to the social consequences of professional decisions in the professions. Many codes of ethics now include commitments to strive for a level of excellence in the practice of the profession. Software engineers are admonished to only advance the profession in a way that is consistent with public interest and to report and failures to do this.

”6.08. Take responsibility for detecting, correcting, and reporting errors in software and associated documents on which they work.” (SE Code, 1999)

The codes of ethics for civil engineers, the IEEE and the various computing societies emphasize "the dignity and worth of other people, personal integrity and honesty, responsibility for work...public safety health and welfare" (Martin, 1989).

Although software engineering does not meet all of these marks of a profession mentioned above (Pour, 2000), it does meet those marks that are significant for understanding how ethics is related to software engineering. There is a special skill and training used to develop products which directly affect many lives and affects the way software artifacts are used in society, and by virtue of that training one gains the power to have a significant effect on society. The responsible application of software engineering principles has a direct effect upon the public's well-being. These two fundamental facts identify software engineering ethics as a professional ethics. Like medical professionals, software engineers are privy to knowledge that is generally inaccessible to people outside the profession. Like lawyers, software engineers have special obligations to clients, obligations that may conflict with obligations to society at large. The difference here is the software engineers owe their primary obligation to society rather than to the client. ” In all these judgments concern for the health, safety and welfare of the public is primary; that is, the “Public Interest” is central …” (SE Code, 1999) Software engineering ethics, though uniquely tied to the technical details of computing, shares common ground with other professional ethics.

Technical and Professional Ethics

The third category in this taxonomy of ethics -- technical ethics -- is different in each of the professions. Although medical ethics and engineering ethics are examples of "Professional Ethics", these two professions are quite different in the way that they implement professional ethics. Each profession operates in a different context with different tools and addresses different problems. Although there are some principles such as "cause no harm" which are true across the professions, in this taxonomy category different professions employ different ethical principles and give these principles different priorities. The physician has a significant obligation to patient confidentiality, but this has little relevance for the mechanical engineer. The technical requirements and skills of the

professional have a direct connection to ethics at this level. The standards and methods to achieve the best medical therapies, the safest bridges, the most reliable software designs are determined within each profession. The intentional failure to follow these standards gives rise to some ethical issues within a profession.

These technical standard and values are determined by consensus within a profession (Gotterbarn,1991;Collins,1994). Society trusts that the educated professionals are best equipped to determine these principles because of their special knowledge. This is the basis for self-regulation of the professions.

Software professionals have a set of standards they can follow and recent work has shown that there is significant agreement about these technical and ethical standards among software professionals. Within the realm of professional ethics there is a surprising degree of agreement (Leventhal et. al., 1992).

Technical Ethics and Values

Although there are technical standards, there are a variety of solutions available for most software engineering problems. The choice of which solution to use is based on professional values. Consequently, professional values have a direct impact on the way we develop applications and the quality of those artifacts.

Software engineering ethics consists of two major elements. One element, called technical ethics, consists of doing a technically competent job at all phases of the software development process, the other element is the use of a set of moral values to guide the technical decisions.

In the professions, our technical skill guides processes: processes of building bridges, healing patients, of building software artifacts. The performance of these processes involves numerous ethical issues. The physician's choice of which medicine to dispense involves a technical judgement about its curative powers, but it also involves questions about the side-effects of the medicine. Is it too expensive for the patient to purchase? Is it addictive? Is it likely to cause other problems? Technical decisions in software engineering are analogous to this. The choice of which life cycle to use may lead to radically different products, although they both versions of the product might satisfy the requirements.

In general, all professional ethics (engineering ethics, legal ethics and medical ethics) are only distinguished by the context to which they apply moral rules. There are, of course, differences in the professional ethics. The contexts bring out different ethical problems. Because different contexts raise different ethical concerns, the order in which the moral rules are applied varies for each application domain. Even within the software engineering process, moral rules are given different significance at different stages of a development life cycle. Consider the different ways informed consent is treated. During the requirements phase of life-critical software, informed consent (understanding

agreement) is an important rule. During testing, principles about not deceiving and not cheating are very important but informed consent is not a major concern.

Although technical responsibility involves a strong sense of individual responsibility, this sense of professional ethics should not be confused with merely following the law. Professionals are aware of many technical solutions to problems. These solutions vary in their adequacy to solve a particular problem. Even if a developed product does not meet the ideological technical goals of the profession there is no cause for legal action until some damage has occurred. There are three basic "requirements for a tort law suit: (1) a duty is owed; (2) the duty is breached; and (3) damages are caused by the breach of duty."(West 1991) Attempts to legally define technical professional responsibility have been unsuccessful. One can be unethical and still not violate the law.

Different values and rules apply to different phases of the software development life cycle and the application of these values in making decisions involves no violation of the law. In the testing phase, if funds are exhausted before the testing is satisfactorily completed and there is no possibility of further funds, there are several options. Whichever option is taken it must be conditioned by moral rules such as "don't deceive", "keep promises", and "act professionally." Depending on the type of software being tested, rules like "don't cause pain" and "don't kill" might also come into play.

In the design phase, other moral rules might get applied first. Consider designing a journal file for a library check-out system to determine the popularity of books and the number of additional copies that should be ordered, if any. The association of the patron's name with the book checked out has a potential for the violation of several moral rules, such as the violation of privacy and the deprivation of pleasure because one does not feel free to read what one wants, and possibly causing psychological pain. Notice that there may be no laws violated by this design even thought there are ethical issues.

In the design phase, the choice of a language for a life-critical system can have moral implications. If the language is hard to debug, write, or modify, then one puts people at risk and violates principles of good system design in the choice of that language. The ethical issues can come from two directions here. If the designers were ignorant and didn't know a better language, they are morally culpable for undertaking a project that was beyond their skill. If on the other hand, they were well aware of these significant differences in languages but decided to stick with the more dangerous language for other reasons such as profit, then they have violated professional ethics. It is not clear that they would lose a tort suit.

Software engineering has many of the marks of a profession and the responsibility of professional ethics. In this sense it is like all other professional ethics. Because the context of software engineering is different from other engineering disciplines, its technical professional ethics is unique.

Technical responsibility is concerned with the approach taken by professionals while they are solving technical problems. Sometimes this is subsumed under terms like 'quality' or

'professionalism'. This involves the consequences of knowing and following good professional standards.

The senses of responsibility that need to be discussed in software engineering include both societal and technical responsibility. Technical responsibility involves moral commitments to standards of software production, to the process whereas societal responsibility involves commitment to using these special skills in the service of society. Software engineers have an obligation to consider the safety, health, and welfare of those who interact with their product. The society which interacts with their products also includes all others participating in the software development process. Software development is a social process and the software engineer has a two-fold obligation to strive for excellence. One source of the obligation is based on technical standards and the other source of the obligation is social responsibility to those who will have to work with his product. Both of these approaches to ethics give positive guidance for our behavior as software practitioners. The activity of software practioners guided by the professional ethical principles is software engineering.

FUTURE TRENDS

The movement toward professionalization of the computer industry and software engineering is continuing. There has been renewed attention to ethical issues within the engineering and computing professions with special attention given to the well-being of the client and the user of software engineering artifacts. The ACM and the IEEE Computer Society adopted the Software Engineering Code of Ethics and Professional Practice in 1998. Since that time it has been adopted by other computing professional organizations and by large and small software development companies as a standard of practice. This has occurred in spite of the fact that in addition to not causing harm, the Code requires software engineers to do what they can to prevent harm. This means that a software engineer is expected to report another software engineer's faulty work. Adherence to this code requires reporting any signs of danger from computer systems, whether or not the individual reporting the risk designed the system.

There are also significant movements toward licensing software engineers and the codification of development standards. Clients are beginning to understand the responsibilities of software engineers and holding them responsible to those standards. The next step in software engineering ethics is a change in culture where there is a mutual understanding and expectation of ethical behavior of software engineers and those who are impacted by their products.

REFERENCES

R. E. Anderson, et al, "ACM Proposed Code of Ethics and Professional Conduct," *Communications of the ACM*, May 1992.

BCS, *A Report on Undergraduate Curricula for Software Engineering Curricula*, The British Computer Society and The Institution for Electrical Engineers, June 1989.

M.D. Bayles, *Professional Ethics*, Wadsworth Publishing Company, California, 1982.

B.W. Boehm, Software Engineering Economics, Prentice Hall, NJ. 1981

B.H. Bauer, "Software Engineering," Information Processing, 71. North Holland, Amsterdam 1972.

W.R. Collins & K.W. Miller, "Paramedic Ethics for Computer Professionals," *The Journal of Systems and Software*, 1992.

W.R. Collins, K.W. Miller & Bethany Spielman," How Good Is Good Enough? An Ethical Analysis of Software Construction and Use," in *Communications of the ACM*, Vol. 37, No. 1 (January 1994), 81-91.

R. DeGeorge, *Business Ethics*, New York, 1982.

G. Ford and N. Gibbs, "A Mature Profession of Software Engineering," Carnegie Mellon University CMU/SEI-96-tr-004)

G. Ford, 1990 SEI Report on Undergraduate Software Engineering Education, Technical Report CMU/SEI-90-TR-3, Carnegie Mellon University, 1990.

B. Gert, *Morality*, New York, Oxford University Press, 1998.

D.W. Gotterbarn, "Computer Ethics: Responsibility Regained," in *Computers, Ethics, and Social Values*, Deborah G. Johnson and Helen Nissenbaum (eds.) Prentice Hall,1994.

D.W. Gotterbarn "How the new Software Engineering Code of Ethics Affects You", *IEEE Software,* November/December, 1999 58-64.

D.W. Gotterbarn (a) "Software Engineering: The New Professionalism," *The Responsible Software Engineer*, ed. Colin Meyer, Springer Verlag 1996

D.W. Gotterbarn (b) "Establishing Standards of Professional Practice," chapter 3 in The Responsible Software Engineer, ed. Colin Meyer, Springer Verlag 1996

W.S. Humphrey, *Managing the Software Process*, Addison Wesley, Reading 1989.

D.G. Johnson, *Computer Ethics* (third edition), Prentice-Hall, New Jersey, 2001.

D.G. Johnson and J.W. Snapper, Ethical Issues in the Use of Computers, Wadsworth Publishing Company, California, 1985.

L.M. Leventhal, K.E. Instone, & D.W .Chilson," Another View of Computer Science Ethics," Journal of Systems and Software, 1992.

C.D. Martin and D. H. Martin, "Comparison of Ethics Codes of Computer Professionals'" Social Science Computer Review,9,1,1990.

M.W. Martin and R. Schinzinger, Ethics in Engineering, McGraw Hill, New York, 1989.

J. Moor, "What is Computer Ethics?," Metaphilosophy,16,4, 1985.

G. Pour, M. Griss, and M. Lutz. “The Push to Make Software Engineering Respectable,” Computer, May 2000, page 35-43

K. Reed, “Software Engineering- A New Millennium?,” IEEE Software July/August 2000 pg 107.

J.H.Schaub and K.Pavlovic,eds, *Engineering Professionalism and Ethics*, John Wiley New York 1983.

SE Code, “Computer Society and ACM Approve Software Engineering Code of Ethics,” *Computer* October 1999, pages 84-88.

J. T. Stevenson, Engineering Ethics: Practices and Principles, Canadian Scholars' Press, Toronto, 1987.

L. B. West ,"Professional Civil Engineering: Responsibility," Journal of Professional Issues in Engineering Education and Practice, 117,4, October 1991.