Student 1 . .

Part la: General Comments

I have mixed feelings about using this code of ethics for software engineering. One hand, most of the imperatives seem to be common sense. Yet this could be because my 'common sense'' is derived partially from the code of ethics I subscribe to as a member of the military. For civilians that may not necessarily share these beliefs, the software engineering code of ethics might not be common sense. I feel that our military-code of ethics has helped develop a level of trust that allows members to function at a higher level than an organization that lacks this trust. The software engineering code of ethics might instill a trust amongst software engineers that allows them to perform at this higher level.

In a detailed examination of the software engineering code of ethics, I noticed several things. First, there seemed to be several instances of reiterating behaviors that are required to be a law abiding citizen in the US. The code of ethics should not attempt to rephrase government laws. Second, many of the imperatives discuss employee/business and client/contractor obligations that do not pertain specifically to software engineering, or even engineering in general. Such imperatives may be part of a code for general employment as a professional, or perhaps employment as an engineer, but they should not be in the code of ethics for software engineering. The discriminator for determining if the imperative should be part of this code of ethics is that it raises the standards of behavior of software engineers above those of the engineering field and professionals in general. In the next few examples, specific imperatives that should be removed or modified are presented.

Part 1b: Identify no more than three individual imperatives that should not be part of code

4.04 'Not knowingly use pirated software on equipment of a client or employer in work performed for a client or employer. " Stating this imperative in a code of ethics is similar to including a clause ' the software engineer must not knowingly violate any federal laws in the use of . . . ' Such an imperative is unnecessary in a code of ethics.

5.07 "Only accept a salary appropriate to professional qualifications.'' This imperative is flawed for two reasons. First, there are no written regulations that relate a salary to professional qualifications. Each organization determines salaries based on a number of factors, only one of which involves the current rate for hiring an engineer with given qualifications. In many cases, the other factors outweigh the qualification factor. Second, the word 11only" implies that the engineer should not accept a fob that pays too much for their qualifications, (em nor should the engineer accept a job that pays too little. Yet the engineer must make this decision based not only on what the qualifications are, but also on personal need: If a software engineer (em needs a job to earn a living, that person should be able to accept the job even if it pays too little.

Likewise, that engineer should not be forced to turn down a job just because it pays too much.

5.10 'Obey all laws governing their work, insofar as such obedience is consistent with the public heath, safety and welfare''. Several problems exist with this imperative. First, I believe the word "laws " should be changed to "rules" or 'regulations''. We are not referring to government laws here - we are referring to the rules and regulations of the organization where the engineer works. Second, it should be common sense that engineers, and professionals in general would not allow rules and regulations to supersede the need for public health, safety and welfare. These needs are generally protected by government laws, and government laws do not need to be restated in a code of ethics.

Part 2: Identify no more than six of the individual imperatives that are aspirational

3.01 "Maintain professional skepticism with respect to any

software or related documents they are asked to evaluate.'' Skepticism cannot be measured. In addition, this imperative implies that the software engineer should be prejudiced, albeit professionally, against work they are asked to evaluate. Perhaps objectivity' should replace the word ''skepticism''. The idea we want to convey is that they engineer examines the software or documents in depth and in an unbiased way, not in a 'this is wrong until you show me otherwise'' frame of mind. This imperative, even if reworded, is intrinsically aspirational.

5.02 'Assure that clients, employers, and supervisors know of this code of ethics''. This imperative cannot be measured on an individual basis. There is no way of measuring how well an individual assured the knowledge of the code to the other people. The action " assure'' does not need to be performed actively to be effective. The organization could however, measure how many people knew about the code. In this case, it would be the responsibility of a particular group of individuals to assure the awareness of the ethics. The effectiveness of that individual or group could then be measured by polling the target population of the organization. This imperative should be removed form the code of ethics and relocated in the " operating instructions" for the group of people performing it. (Note: This imperative is similar to the statement -ensure the occupants of the aircraft know the proper escape procedures" - it must be performed by the staff of the aircraft, not by each passenger.)

5.03 and 5.09 also cannot be measured. The problem is caused by the verbs -Support" and -Help". Support cannot be measured in any meaningful units, and neither can help. These are truly aspirational imperatives, but unlike 5.02, they should be performed by individuals. Therefore they should remain in the code of ethics as aspirational tenets.

5.12 "Improve public knowledge of software engineering." It would not be feasible to measure the effect of an individual's contribution on the improvement of the-public's knowledge" of software engineering. This imperative is also aspirational in nature, and it cannot be rephrased in a way that would make it quantitatively measurable.

Student 2

1. Critical Evaluation

a. General Comments about this Code

Generally, this code is a good start in defining the bounds of our blossoming profession. The seven rules with their associated imperatives address the various areas where our profession's ethics come into play. I certainly believe there is room to expand this list, but I would hope that expansion will happen with maturity. The important point here is that an essential step has been taken that serve as a foundation for strong ethical standards in software engineering. This foundation can do nothing but improve the profession as a whole.

b. 3 Imperatives to Remove

- 1.12. is far too broad in scope to be considered a imperative in this list. I believe each imperative under every broadly defined rule should define individual important areas of that rule. The imperative 1.12. can be broken down into potentially 5 imperatives that further define the scope. For example, one imperative can discuss the importance of identifying, defining, and addressing environmental issue with more specifics on what kinds of general environmental issues are involved.

- 5.02 is perhaps a bit overkill. We should strive to advertise our ethics aggressively internal to our profession. However, following an imperative that requires us to explain our code of ethics to all clients, employers, and supervisors could be very difficult to achieve and, T believe, largely ignored. This imperative could be "watered down" a bit by encouraging external advertisement of our ethics or externally advertising only that we have a code of ethics.

- 5.11. is far too vague to be an imperative. Its unclear if this imperative is encouraging software engineers to participate in civic affairs or to serve constructively when participating in civic affairs. Also, this imperative doesn't indicate which civic affairs a software engineer should be serving in. Finally, I question if this imperative is supposed to be under Self rather than Professional if it should be imperative at all.

c. Importance of Ethics for Groups

Of course ethics is needed in groups, particularly groups that have such a drastic effect on people lives like software engineers. The most important reason is trust. If the public knows a group has and practices good ethics, the public will tend to trust that group more. This can clearly be seen in the profession of politics in the last 30 years. Flagrant violations of laws and basic ethics, from Watergate to Iran-Contra, had steadily eroded public trust for our political leaders. We can learn from these mistakes to build trust with our profession and avoid the pitfalls that come from a lack of ethics.

2. Aspirational Imperatives

- 1.10. is a vital imperative but is a very difficult imperative to measure. The law might be one place to determine whether or not privacy is being violated, but privacy issues have been a real legal battleground lately. I really believe this aspirational imperative should remain aspirational, allowing it to survive as privacy issues change and mature.

- 9.01. brings up the question, "What is competence?" I may feel as a trained software engineer that I have the competence to take on real time system project even though I've never worked on a real time system. Competence seems to be a term open to interpretation, making it difficult to measure. I suggest eliminating this imperative and letting the imperatives under Self and Profession carry the meaning embodied in this imperative.

- 3.01. is like 9.01 in that skepticism is a term open to individual interpretation. This can be changed to read, "Provide honest, critical evaluations of software and related documents."

- 5.01. is another hard to measure term open to interpretation. An improper use of this ethic can in itself be unethical. For example, if you deem a company unreputable and deny that company your business on a gut feel rather than hard evidence. You could alter this imperative to focus on businesses who have been convicted of illegal practices.

- 5.03. should read the other direction, "Don't support SE's who are found in violation of this code." The number of SE's who are convicted of violating the code are more easily measurable and easier to identify.

- 6.13. again brings up the problem of interpretation. What one person thinks is fair and just compensation doesn't necessarily hold true for another. This is another imperative that I thinks should remain aspirational in the code. It may be difficult to measure, but its always a good value to shoot for.

Student 3

a. General Comments

I think the intent of the code is good. But I strongly disagree with any attempt to "legislate" what is really common sense. Thus, I feel the SW/E " code" should be less detailed, more like the 10 Commandments. Or, perhaps a more appropriate example, the Hippocratic Oath. I have attached a hard copy of a web page that provides a contemporary review of the Oath, and several humors secular variations. My point is, in the various oaths used by the medical community, roughly 5 to 10 statements are sufficient to " guide'' the profession. Frankly, for the software professional, I would be happy to see just the seven rules in the current draft be the "Ethics of the Software Professional" .

b. Definite Don't Includes

Definitely, 1.19 should be eliminated. By saying that someone should depart from standards practices only when justified" is a hunting license to depart from standard anytime you can justify it. Sounds like an Ada waiver.

Another is 1.03. One, let's be honest, while we would like to think that education alone is sufficient to qualify someone, reality is, without experience, the individual is still ill prepared. So, how does one become experienced if they can't practice without being experienced? Perhaps reword the clause to read ' ... for which they are responsible''.

My third candidate is 6.11. -Not supplant another software engineer after steps have been taken for employment.'' What in the world does this mean? I looked up supplant, it says - to be replaced by another" . Ok, this still doesn't make sense.

c. My "Lieutenant " Feelings on "A Code of Ethics"

My personal view on this issue is biased because I'm a member of the Armed Forces. We operate under this little thing called the UCMJ; very Enforceable then shahs and the, shall net,. Interestingly enough, it is my opinion that the "rigid" military environment actually gives us more latitude in our endeavors because we know the rules. In the private sector, there are just as many limits on one's conduct, however, most of them are implied. Of course I'm speaking in general terms, not specifically with regards to software activities. Reinforcing my position from part (bf a using the military example, our Oath is just a short paragraph, but it total defines what we are and how we are to do it!

"Aspirational " verses ' ., embodied in a code of practice."

Here I'm going to use your own words, not against you necessarily, but to further emphasize my theme. 7 believe what this committee has begun is more a enumeration of the practices of the profession of software engineering, i.e. [em code of practices, than a statement of what is hem ethical (right verses wrong) with respect to software engineering. If its ethics we are defining, then indeed, they should be aspirational. Again, I propose the seven rules (sounds like Covey) as the [em Code of Ethics to provide the vision and direction to the profession. Under that I would put the other subsidiary clauses as hem Examples of practices. Even the Introduction states 11 no set of subsidiary clauses exhaust the general rule." So since there is no complete code of practices, why try to legislate it.

Since flexibility is the key to Airpower, I've answered this section in my own way rather than providing six examples. I'm sure there are enough sensates (Myers-Briggs Type Indicators) in the class to provide you with a laundry list.

Student 4

1. Critical Evaluation of Software Engineering Code of Ethics

General Comments on IEEE/ACM Code: To be a profession, an occupation has to have some attributes which discriminate ii from non-professions. A code of ethics would go a long way towards discriminating Software Engineering from software hacking.

Since I personally have no experience with other engineering professions' codes of ethics, I have nothing to compare this code to. I feel, however, that much of what is here is a "shrink-wrapped" mix of common sense and "motherhood" style aphorisms.

Overall, I'd structure the code into two sections: Imperatives that are testable, and thus measurable and enforceable; Imperatives that are worthy goals.

Remove these imperatives: (most others are useful things we should strive to do, be.)

1.19 Avoid Fads, Stick with Standard Practices: What's a Fad? Should we have avoided the entire 00 "fad"? Should we have eschewed the Client/Server movement? It seems to me this imperative would only stifle creativity and innovation. Besides, what are the standard practices? This field has evolved faster than any other--Standard Practices of today could be outdated and inefficient in two years.

9.01 Provide Service only in areas of competence. First, how do you know if you're competent? By taking the standardized test?(HA).

Second, if you only work in areas you are competent in, then how can you ever hope to get a job doing something else, and how did you achieve competence in the first place? Award of a Bachelors or Masters degree is conference that you understand certain ideas and material--it does not mean that you're competent (hence the separate test to be a P.E.). Once again, since software is generally not based upon principles of the physical world, I submit that it would be nearly impossible to create a decent test of competency.

5.05 Report violations of this code to appropriate authorities. "I'd like to report that Joseph Blough does not understand fully the specifications for the software he is working on." Silence, followed by laughter on the other side of the phone line. This imperative should be removed completely, or .. 1) restructure the code of ethics into testable and aspirational parts 2) make 5.05 apply only to the testable parts of the code.

Comments on applying a code to a group of individuals Since most of these imperatives are aspirational (see below), rather than strictly testable, there really isn't a problem applying this code to a group of people.

Oh the other hand, if everything was strictly "testable", then applying the code would also incur enforcing the code, which means there would be punishment of the guilty, and a consistent means of determining who's guilty and who's innocent.

Now, we must ask ourselves the question, "Could an American Software Engineer follow this code without being ostracized from the dirty, no-holds-barred, cutthroat, dog-eat-dog world of American Business?" I don't think so; Clark Kent gets by with his forthright attitude because he has magic powers the rest of us mortals don't have. I'm not saying that we shouldn't have a code of ethics, I'm just saying we can't expect everyone to follow it. I'm not going to tell my customer his project is going to turn out badly (assuming my management won't do it) when it's going to take food out of my children's mouths. (I might start looking for another Dab with a reputable company though).

2. Aspirational Imperatives

It seems to me that almost all of these imperatives are aspirational, except for the ones that can be verified In a binary manner, e.g., you either accept or reject a bribe. That doesn't mean that they're not important--it just means that these imperatives reflect the truth about our fuzzy, gray world (that is, ahem, that it is a fuzzy, gray world)..

Here then are some aspirational imperatives and what to do with them.

1.01. Assure specs are good...
HA, this is the "holy grail" of software engineering. If we could Assure specs were good in the first place, then we wouldn't have to wrestle with many of the problems of quality, timeliness, and cost that we have.

What to do? Nothing! Recognize this aspiration for what it is!, an aspiration, a worthy goal. We should all strive to ensure specs are "good"...

1.02. Assure engineer fully understands the specs...

This aspirational imperative is nearly the same as 1.01. as above.

5.07 Only accept a salary appropriate to professional qualifications
A problem for several reasons.
1. It's hard to quantify/verify a professionals' qualifications.
2. What does "appropriate" mean?
3. It acts as though professional jobs are monolithic (generic software guy), yet there are many kinds of software jobs, each with different pay scales (MIS vs. Embedded Software Engineering vs. Software Test.

This should be restated as something like... "Don't accept a salary more than 108 lower or 108 higher than the median in that local market for a position with similar responsibilities."

Student 5

In reviewing the software engineering code of ethics, I found it to be an almost stifling document, filled with grandiose ideals of what a software engineer should be. Most of the concepts that it embodies can be applied to any professional group or affiliation with minor changes; what's missing, for the most part, are areas specific to software engineering. I find it hard to believe that such a verbose listing of "imperatives" is necessary; perhaps it's just me, but I find this almost insulting to my sense of propriety, as if the author is questioning my values by listing things which should clearly be innate.

Unrealistic

1.13: "Promote maximum productivity and minimum cost to employer, customer, user, and public."

Does this really belong in a code of ethics? If society has regressed to the point where we actually need to codify ideals such as this, then this "code of ethics" would need to be volumes larger. Note that this applies to a great many of the imperatives listed as well; is it really necessary to list that which should be a given?

5.01: "Associate only with reputable businesses."

Again we have a question of semantics-what marks a business as "reputable"? In any case, with what businesses a software engineer chooses to work should not be in the purview of a professional organization.

5.07: "Only accept a salary appropriate to professional qualifications."

Who decides? By what measure is one's qualifications (and by extension, one's salary) measured? If your employer values your skills that highly, why should you not accept the salary, no matter what level?

Aspirational

1.03: "Assure than they are qualified, by education and experience, for any project on which they work"

I feel that this imperative is a bit limiting; after all, how is a person to grow, to improve, unless they take on such challenges? I agree that projects that are obviously out of reach should be avoided, but if this were to be followed, I doubt that many people at AFIT would be qualified for the thesis they are working on.

1.07: "Assure proper estimates of cost, schedule, personnel, and outcome on any project on which they work."

Again, nice but not really attainable. Software development is by nature a rapidly-changing project; to define "proper estimates" ignores the reality of how things need to adapt to an ever-changing situation.

1.14: "Avoid fads, departing from standard practices only when justified."

What is the definition of "justified"? I'm sure that when object oriented programming was first envisioned, it was probably considered a fad. There will be times when non-standard practices are optimal in a given situation, but what defines the limits of justification?

5.03: "Support software engineers who do as this code requires."

Here we see a disturbing concept: “ as this code requires.” Does this then mean that we are too shun those who seek their own path? This could probably be better worded along the lines of "Support software engineers who strive to attain the ideals of this code", but that is pretty reaching itself.

5.11: "Serve In civic affairs constructively."

Huh? Why should a professional organization concerned with software engineering be interested in how one serves in civic affairs? Granted, members of this organization should strive for impeccable public character, but this is beyond the scope of this document.

Student 6

la. I have mixed feelings about this document. On one hand, I believe it presents many good general guidelines to follow. Many of them are common sense, and most, if not all, are things that should be second nature to officers. My big complaint about the code is the lack of emphasis on the need for at least a high level understanding of the environment the software the engineer is working on resides. This environment includes such things as the surrounding hardware pieces, networks and their corresponding interfaces, and other various system level issues. I think this kind of information is critical for software engineers to understand, and I don't see any kind of emphasis on that here. It seems like the software world sometimes has a tendency to stick its head in the sand and focus only on software, ignoring all the outside issues which frequently can doom a project to failure if not sufficiently considered and understood.

1b.

Item 1.13: Promote maximum productivity and minimum cost to employer, customer, user, and public. I don't agree with this one because I don't believe the "minimum cost" of doing something is always the best way. It's ok to present that option, but I think too often we end up buying a "cheap" short-term fix, which ends up being very expensive in the long term.

Item 9.02: Assure that any document upon which they rely has been approved by someone qualified to approve it. While this sounds good, I don't

think this item is realistic. It places the responsibility for accuracy/veracity on the engineer, rather than the producer and/or manager of the document. The engineer should definitely use their innate "engineering intuition" filter when reviewing any document they need, but there are many times (from personal experience) when documents I've reviewed look right/sound right, unfortunately they are several revisions old. You do learn after a while which documents to trust and which ones not to, but ultimately your documents are only as good as your document manager is. Nor does a software engineer typically have time to personally validate each document he needs to do his job.

Item 5.07: Only accept a salary appropriate to professional qualifications. This should be dictated by the market, not by the criteria above. The criteria listed are subjective (good qualifications to one person don't cut it with another) and salaries vary widely by region and industry.

lc. As already alluded to in section la, I believe most of this (particularly the portion emphasizing ethics) should be second nature to officers. Given the current state of ethics in the civilian community, I believe this code could be quite eye-opening.

2. a. Item 1.01: Assure that specifications for software on which they work have been put in writing, satisfy the user's requirements, and have the customer's approval. I believe this item is not realistic. I think this is a laudable goal, and in some cases I think this can be done, but in all lobs I have held, the requirements are fluid. In the satellite operations world, requirements change because the operational environment changes (satellites fail or degrade, orbits change, new satellites are launched, special missions are required, etc.). I think this item should start with a "When possible ..... .

b. Item 1.08: Assure adequate documentation on any project on which they work, including a log of problems discovered and solutions adopted. Adequate documentation is not just nice to have, it can often be crucial to working a problem. That said, however, the term "adequate documentation" Is nebulous. I believe it would be helpful to have engineers use the processes they have in place. For example, make sure they update programmers notebooks, follow their organizations programming style (particularly with respect to in-line documentation), make the appropriate manual changes/updates as their process requires, and make sure any problems are reported using the organizations problem reporting procedure. Perhaps this item could emphasizing utilization of their organization's documentation processes.

c. Item 2.03. Affix their signature only to documents prepared under their supervision and within their areas of competence. While this is a laudable goal, many times it is impossible to do, particularly for an Air Force officer. Speaking from personal experience, if I waited to sign off on a document before I understand every nuance, I wouldn't have signed many documents. I believe its important for an engineer in charge of a large project to find people in whose expertise they trust when the project expands beyond their area of competence. At minimum, this item needs a "Whenever possible, ... As with many of these items, a little common sense (and sound engineering intuition) goes a long way.

d. Item 9.01. Provide service only in area of competence. This is certainly desirable, but many times its not possible. Many times, software engineers are forced to work in areas right on the edge (or even outside) their areas of expertise. This item would be better if it said "Whenever avoidable, do not provide service outside your area of competence."

e. Item 5.01. Associate only with reputable businesses. While I don't disagree with the spirit of this item, some problems can occur if strictly followed. Occasionally, you will find that a parent company may be considered rather suspect, while the local division of the same company upholds high standards. The opposite can occur as well. I've been in situations as well where the worker bees of the company did a fine job, but unfortunately, the company management folks weren't exactly honest w/the Air Force, thereby causing lots of problems. Finally, we (as officers) are often placed in situations where we must work with a disreputable contractor due to contractual obligations-- in this case we must make the best of a bad situation.

f. Item 5.11. Serve in civic affairs constructively. This item is too vague. If the intent is to present a positive public of software engineers to the public, then some suggestions should be included in the item. Some examples could be high school outreach programs letting students know what kind of opportunities software engineering has to offer, serving as Judges in science fairs, etc.

Student 7

1. General Comments

In general, I found the Code of Ethics for Software Engineers to be a good document. It expresses the need for ethics and addresses the areas where ethical behavior must be applied. I do disagree with the level of detail of some of the imperatives. Rather than pertain to ethical practices, some imperatives seem to delve into job performance. For example, Rule 5 is a good statement that I interpret to mean that software engineers should act with integrity and good conduct. Specifically, imperative 5.13 discusses the need to share lob knowledge with other professionals in a variety of forums. I do not think my participation in conferences or publication of articles is an ethical issue. If the intent is to say that I should not keep secrets about new discoveries in the area, it is justified; but the way it reads someone could interpret it to mean that they must participate in every possible forum to be ethical. I do not think the later is an ethical issue, but one of professional choice.

2. Imperatives that should not be part of this code

1.14 Avoid fads, departing from standard practices only when justified.

I do not see the connection between fads and non-ethical behavior. Sometimes, good ideas start out as fads and are rejected by the establishment at first. Only after proven use by those bold enough to go against the establishment are these accepted. This imperative seems to quell some latitude for creativity which is greatly needed as this infant discipline searches to define a majority of its aspects. Amore ethical statement could be to not institute new or faddish practices without first discussing them with an employer or client.

4.01 Provide service only in areas of competence

With the amount of downsizing, I am sure most Air Force officers as well as employees of civilian corporations have had to accept responsibilities that are somewhat out of their area of expertise and competence. However, as professionals, we have the responsibility to do our best job and learn as much as we can to perform the new job.

This imperative seems to say, unless you already have the knowledge, do not take on anything new. This could stunt the professional growth of software engineers. I think a better statement may be to appraise an employer or client of areas which are new or beyond your experience to date and let them make the decision if they want you to pursue it or not.

5.12 Improve public knowledge of software engineering

Although important to furthering the discipline and encouraging young professionals to enter, this imperative has no pertinence to ethics. If I do not speak publicly about the merits of software engineering, have I violated an ethical premise? In my opinion, no. An uninformed public may be resistant to software engineering practices as a result, but I am not acting in an unethical manner.

3. All six of the imperatives that I feel are aspirations fall under rule 1.
They are:
1.02 Assure that they understand fully the specifications for software on which they work

How do you measure full understanding of the specifications. I can begin work thinking that I fully understand something but maybe about halfway through I realize that my understanding was based on an erroneous assumption. One person may interpret full understanding to be flawless understanding while another may interpret it to be the best understanding possible given the information provided. In either case an individual's understanding is difficult to measure. Reword - Discuss specifications with employer or client to insure both parties are in agreement.

1.09 Assure proper goals and objectives for any project on which they work

Who defines proper? Once again, different people may interpret this in different ways and it makes measurement difficult. Reword Discuss goals and objectives with employer or client to insure both parties are in agreement.

1.05 Assure proper development methodology on any project on which they work

Since development methodology is still non-standard in many cases, who determines what is proper and what is not? Reword - Apply justifiable development methodology to any project on which they work.

1.06 Assure proper management on any project on which they work

Once again, who determines what is proper? Reword - Apply justifiable management techniques to any project on which they work to insure quality control and risk are accounted for.

1.08 Assure adequate documentation on any project on which they work

Who determines adequate? What is adequate to me may not be adequate to someone else. Another very subjective item. Reword Discuss documentation required with employer or client to insure both parties are inn agreement.

1.09 Assure proper testing, debugging, and review of software and related documents on which they work

Once again, the word proper begs the question of who defines what is proper. Reword - Apply justifiable testing, debugging, and software reviews to the products on which they work.

Student 8

General Comments:

In summary, I am distressed that any organization should be in need of a code of ethics aside from the original Code of Ethics that has been with us all the while. Why is it that we need a separate document to tell us that we should not accept a bribe, or to always do our best? Naturally, the original Code of Ethics I'm speaking about is the Bible. It seems demeaning in some way to have any professional organization espouse a written list of ethics (thereby intoning that there are those who would not hold to those ethics if there were not written down) that I have taken for granted nearly all my life.

Does this mean then that this particular code is houseful? No, I don't think so. Even as we have found in scripture that well intentioned people go astray, so there is need to write down a list of rules for each of us to obey. Why else did God give the people Israel the Ten Commandments?

My general contention is that the moral base of our professional society is slipping to the point that we must be reminded that "Thou shall not steal...

As for the actual wording of this particular code of ethics for software engineers, it seems that a lot of work has been put into it to describe the highest qualities of professionalism that software engineers need to display.

It should go well in helping to establish software engineering as one of the foundational engineering practices.

What Should Not Be There:

3.02. As stated above, I am quit document of this call],,, that would not ASSUME the bribe.

It just seems to me that these rules could be built with another set of rules in mind.

Is this realistic? Probably not. I am well aware of our pluralism in this day and age, and that a shrinking majority of people have in mind this same set of ethics \ that I speak. I am just generally bothered that this must be in here.

Aspirational:

5.03. It seems that "Support software engineers who do as this code requires." is a bit aspirational. What does this really mean? What if I disagree with another person professionally? What does it mean to support them as this code requires? How do I measure that? It seems pretty vague.

Student 9

1.

a. In general a code of ethics should be adopted by any group who wishes to maintain an ethical approach to business. In the most general case most ethical decisions seem to be common sense but in the case of someone who was not raised in an ethical manner these rules should probably be spelled out for them. For those people who have been raised ethically this code of ethics could be insulting unless they look at the fact it was written to inform those who have no ethics.

b.l. 1.10 How can documents interfere with the privacy of those persons reading said manuals. This seems ridiculous to think that a written document can intrude on the privacy of an individual.

b.2. 3.06 Seems shaky due to the fact that it contradicts the negotiating process of suiting a potential client with the proper software for the job the client is requesting. It almost seems as if a hands-off policy is adopted after a client solicits the company. Essential communication between those working on a project and the buyer of said product should ensue so that those paying for the software get the desired product.

63. 5.03 This statement in my opinion potentially asks unethical people to support ethical people and it seems limited to only Software Engineers who should be adapting to this code of ethics since it was written specifically for them. In general this statement seems ridiculous.

c. Hopefully most individuals with a college education have some form of ethics. Unfortunately some schools allow unethical behavior and some of those people without ethics will make it into the professional world. If these ethics are so valuable to the profession they have been written for, These ethics should be able to be enforced and those members not abiding to the ethics set forth in this document should be booted from the profession until they prove in one form or another they have obtained an ethical attitude toward the standards which have been set.

2.

a. 9.01 Sometimes people need to learn a job before they can do it. As long as this is understood then a person who is not fully competent in a field may gain work experience in order to get this job done. In other words... not everyone is an expert and training is required.

b. 5.09 In the world of the government how do you reorganize an organization to be favorable to acting ethically since most organizations are formed in the government to save highly paid officials the rate of pay they are currently earning.

c. 5.05 Who are the proper authorities? Since there is no certification for software engineers what actions can be taken against people who do not conform to these ethics.

d. 5.11 This seems like a constitutional violation of the first amendment (right to free speech) People should be able to act "nonconstructively" in civic affairs as long as they protest peacefully.

e. 6.02 Seems difficult to maintain since sometimes a boss asks for reviews of other peoples work and you may be selected to do that. Thus you will be reviewing other peoples work possibly without their knowledge.

f. 5.01 This may be hard to identify unless a full search is done by an individual before doing business with a company. Once a contract is signed the contract should be binding.

Student 10

l.a. Overall, the code is pretty good. On a scale from 1 to 10, 1 would give it a 6 or a 7. I think the 7 rules hit all the main areas of concern for any field. Software Production or something similar as it extends beyond the product to the aspects of making the product and managing the effort to make the product.

The major negative to the document is the number of "weasel words" used in the imperatives. A "weasel word" is a word meant to give you leeway in interpretation and measurement. Some examples from the code are "proper," "appropriate," and "badly." These type words are not always inappropriate (oops!), but in many cases detract from codifying or making concrete the type of desired behavior. Most of the my selections for part 2 relate to the overuse of "weasel words."

l.b. I believe the following imperatives, as written should not be a part of the code:

1.19 Avoid fads, departing from standard practices only when justified.

This imperative is well-intentioned, but suffers from two flaws, one of which is fatal. Namely, the identification of fads can never be done with certainty. A good example is using lines of code to measure software size. In a recent presentation at the Dayton Marriott, Capers Jones asserted this is malpractice, yet a large contingent of industry has used, does use, and will use this measure. Is LOC a fad? Or, is it stepping stone to better measures that industry will eventually outgrow?

9.09 Represent no interest adverse to their employer's without the employer's consent. Even when couched in the context provided by the parent rule, this imperative is too encompassing. First of all, it assumes the employer and/or client are basically "good" and follow some code of ethics, which in most cases, probably is not true. Also, it seems to prohibit actions such as lob-hunting (without first informing your employer AND getting permission), investing money in a company that in any way competes with your employer, or opposing your employer's business plan for any reason other than those given in Rule2.

1.13 Promote maximum productivity and minimum cost to employer, customer, user, and public.

I'm not as hard over on eliminating this one, but I don't think promoting minimum cost to the customer, user, and public should be included. The customer, user, and public have the right to be safe using your product, but have no rights concerning the price they are will to pay for your product. I could live with a weasel wording such as "fair price."

l.c I feel the definition/use/application of a code of ethics is crucial to creating an occupation such as software engineering. What binds the United States together in a general sense is belief in such ideals as freedom, democracy, and justice. Such ideals are needed to bind professional together into an occupation or profession. The exact implementation of those ideals change, as do the laws of our land, but the ideals themselves remain hard and fast. This code of ethics is a first cut at the laws of the land. It is an important and necessary step in creating the discipline of software engineering.

2. The following imperatives can not be measured in the present form:

1.02 Assure that they understand fully the specifications for the software on which they work.

Aspirational.

The real measure of understanding the specifications is to build the system. Short of that, there could possibly be some rating or assessment by the customer, but that would be very subjective.

1.09 Assure proper goals and objectives for any project on which they work.

Reword to emphasize alignment of goals and objectives with specifications of the product; expectations of the customer; and goals, objectives, and expectations of the employer. e.g. "Assure goals and objectives for any project on which they work, align with the expectations, goals, and objectives of the customer and employer." This assumes the customer and employer have such expectations, goals, and objectives (a reasonable assumption).

1.06 Assure proper management on any project on which they work, including proper procedures for control of quality and risk. Some more words to elaborate "proper" management are needed. Proper procedures for quality and risk are not enough. Should include management of "*ileitis".

5.01 Associate only with reputable businesses.

Reputable is pretty hard to measure, but should at least include some examples of non-reputable businesses, such as those involved in criminal activities or those whose members violate the imperatives given in rules 2 and 3.

5.11 Serve In chic affairs constructively.

Aspirational.

"Constructively" should be explained and should be more quantifiable, if possible. However, this one seems to be a "know it when you see it" type.

NEW ONE

1.15 Assure the quality of the product meets the quality requirements of the user and/or employer.

Student 11

Generally, I liked this Software Engineering Code of Ethics. The Code of Ethics seem to embody the ideas of integrity, honesty, and professionalism. For the most part, the rules and imperatives are based on practicing good common sense and judgment that everyone should do daily. The three imperatives that I did not think belonged to this Code of Ethics are 9.01, 5.07, and 5.11. I think that the basic concept underlying 9.01 is achieved when all the other ones are followed. Also, one may just be learning a new area of expertise and will gain more competence with time. Imperative 9.01 seems to discourage trying to venture out and explore new ideas. I do not think imperative 5.07 should be in this code because fairness of salary is subjective. Employers and employees usually seem to differ in what they consider is appropriate therefore, it should be worked out between them and not be mandated by code. Perhaps it was the phrasing but imperative 5.11 seemed out of place in the list for rule 5. 1 think that as good citizens we are responsible to deserve in civic affairs constructively but this in no way effects us as good Software Engineers. I think that having a Code of Ethics is good because it gives people known standards that they must all abide. I think that the usefulness of the code really depends on individuals since no one monitors them. Each and every one is ultimately responsible for practicing their own ethical day-to-day behavior.

The six aspirational imperatives I found are 1.09, 1.10, 1.13, 3.01, 6.09, and 6.13. Three of these imperatives cannot be reworded and are purely aspirational. The first 1.13 could not be change because I think trying to maximize productivity and minimize cost can be achieved in numerous different ways specific for each project. I think that the only way to capture is intention to apply to everyone is to keep it aspirational. The next imperative 6.09 is also aspirational. I think that every company has its own way documenting criticisms. Furthermore, criticizing someone is highly subjective and can be viewed as fair by one and bias by another. This makes it hard to quantify. The last imperative 6.13 is another idea that is hard to define as concrete set of activities which can apply to everyone. Fair and just compensation is interpreted differently by everyone.

I think that the other three aspirational imperatives can be reworded. 1.09 can be changed to Assure that the objectives and goals set for each project address needs of the projects customers. The word proper in the original wording is too subjective. For 1.10, I think that the wording should be changed to Assure that software and related documents on which they work respect the privacy of those subject to the software by informing and gaining permission whenever Necessary. This gives a way to ensure privacy to safeguarded. Then, for imperative 3.01, the wording should be maintain honesty and professional objectivity against outside influences when evaluating any software or related documents To me this is better defined than the original phrasing.

Student 12

l.a. General Comments on this Code of Ethics.

I think it's an excellent idea and a vital step in establishing software engineering as a discipline. The code establishes a baseline for ethical activity within the software engineering community and serves to "legitimize" the profession to those outside the software engineering community.

1.6. Three imperatives that should not be part of this code of ethics:

9.02 Assure that any document upon which they rely has been approved by someone qualified to approve it.

This places too much of the burden on the person relying on the document and not enough responsibility on the person that approved the document. Also, this should be covered by the approver's ethics under 2.03 (Affix their signature only to documents prepared under their supervision and within their areas of competence.).

5.07. Only accept a salary appropriate to professional qualifications.

What does this mean? I have always been leery about the use of the terminology "appropriate salary". Who is to say what is appropriate? The market should dictate how much (or how little) software engineers, basketball players, and AFIT instructors should make. I would not consider it unethical to be paid more for performing the same ethical work as my counterpart in another company-- I would consider myself lucky and worth every penny.

5.10. Serve in civic affairs constructively.

"Constructively" according to who? Assuming civic affairs means your conduct as a citizen in the community, I disagree. Your ethics in your lob may very well overlap into your private life, but it should not be part of the code of ethics for your profession. This is a privacy issue. If Joe (or Joanne) Software Engineer behaves ethically in his/her profession, yet is an active member of the Communist Party and hopes to replace the present government with Communism (which by some, would certainly not be considered constructive), so be it. This should not be considered unethical, as far as his/her professional conduct as a software engineer.

l.c. General feelings on definition/use/application of any code of ethics:

I think it's not only practical, but necessary to have a code of ethics documented for a particular profession. For some people, ethical behavior is innate and they would adhere to the code without even knowing of its ...stones. They would only need the code to specify the particular application of ethics to what they do in their job. For others, although they may be expected to engage in ethical behavior most times, they must have things "spelled out" as to what ethical behavior is in their line of work, so no claim can be made that, "I didn't know...". Unfortunately, there are still others that cannot be expected to behave ethically, and a documented code (that they have ascribed to formally by literally signing up to it, or informally by just being a member of the profession) serves to stipulate the particular case when a member of the profession violates the code by behaving unethically.

Any profession that directly or indirectly has an impact on the health, safety, and welfare of people (which software engineering does, without question), should have a code of ethics at its core. If you had a nuclear reactor being built in your backyard, you would surely want the people designing the structure to behave ethically, so as not to endanger anyone. The same is true for the people that develop the software to run the nuclear reactor-- there is no difference between (preventable) substandard design or use of materials that cause a catastrophe and a (preventable) error in software that causes a catastrophe.

As with all codes of ethics, this code could be replaced by a strict adherence to the "Golden Rule".

2. Six "aspirational" imperatives:

1.03 Assure that they are qualified, by education and experience, for any project on which they work.
(can be nothing more than aspirational)

Qualifications are important to work on any project, and software engineers should avoid working on projects when they are not qualified. I take this to mean a project where the software engineer is clearly "in over their head". There is a lot to be said about on-the-job training and "experience", where the software engineer becomes qualified to work on such a project only as a result of working on the project.

1.09 Assure proper goals and objectives for any project on which they work. (can be nothing more than aspirational)

This is a matter of perspective, focus, and definition. A software engineer working on the details of a system may not know about or care about the goals or objectives for the overall system. In addition, these overall goals may detract from the sub goals he or she needs to concentrate on. Finally, the term "proper" is extremely subjective. (The term "proper" is also used on 1.05, 1.06, 1.07, 1.09-- 1.08 uses another subjective term, adequate)

9.07 Inform client or employer promptly if, in their opinion, a project is likely to fail, to prove too expensive, to violate copyright laws, or otherwise turn out badly. (split and reword, "turn out badly" is much too subjective)

9.07.1 Inform client or employer promptly if, in their opinion, a project is likely to fail to meet schedule or budget constraints beyond prescribed tolerances.

9.07.2 Inform client or employer promptly if, in their opinion, a project is likely to violate established law, including: copyright laws, patent laws, trademark laws, etc.

5.01 Associate only with reputable businesses. (can be nothing more than aspirational)

Major Kanko, are you now or have you ever been a member of the Communist Party?
While 5.01 sounds like a great idea, how could you possibly enforce it? Who defines what's reputable? If the CEO of a company gets a DWI, is that good reason not to associate with the company any longer? This could possibly be reworded to state, "avoid businesses that have documented cases of unethical behavior", but it would still be nothing more than inspirational, because of the subjective terminology.

5.09 Help develop an organizational environment favorable to acting ethically. (can be nothing more than aspirational)

With terms such as "Help", "environment", and "favorable", this is impossible to test.

6.01 Assist co-workers in professional development. (can be nothing more than aspirational)

This would be quite an accomplishment in practice, but can be nothing more than aspirational, since assistance could only be encouraged, and can't be measured.

Student 13

In general, I approve of a code of ethics for software engineers. I believe that ethics are an important part of any profession because it sets a standard of conduct that members of the profession should be able to rely on. I, personally, would like to be able to rely on the ethical behavior of my employees, my professional associates, my fellow workers and my employers.

Whether a code of ethics becomes part of a certification process for software engineers or not it will still be useful as it provides a forum for discussing and debating ethical issues related to our profession. Even if Software engineers never have a certification process, individual employers or professional societies may decide to use it as guidelines/rules for ethical behavior within

It is important in all levels of society that `appropriate' behavior be the norm.

In many groups behavior standards for attitude, work habits, appearance, language, dress etc. are defined, trained and enforced. Ethical behavior standards can be fuzzier than other types of behavior standards yet they, in my opinion, form the foundation for all other standards. Ethical behavior standards need to be defined, trained and enforced. A code of ethics is an important part of defining ethical behavior standards but it is not the only part.

The specific code of ethics used in the homework 2 handout, in my opinion, has problems. I see no real problems with the 7 Rules. I would change or eliminate many of the specific application general problems with them is that 1) There are too many of them. 2) They are inconsistent in form/meaning somehow. The fact that prescriptive and aspirational statements are intermixed could be part of the problem. They also appear to be written by different people at different times for different reasons.

In particular, I found many of the application statements (only some are listed) fell within the following problem classes:

Some application statements... 1) do not belong in the code of ethics or are out of place (1.19, 6.07)

2) are too vague and could be interpreted in several different ways (1.09, 1.19, 6.11) 3) are redundant and restate other statements in a more specific or more general way (1.08, Most of 3.0x) 9) concern issues outside of ethical conduct (5.11, 5.12) 5) are too emphatic about a generalized issue which is not black and white in many cases (1.03, 1.10, 2.06, 5.07, 6.08) and some were in more than one of the above classes.

The following statements do not belong in the code of ethics. Statement 1.09 "Assure proper goals and objectives for any project on which they work" falls into categories 1, 2 and 9. Is software that controls a nuclear missile `proper'? What about software that aids in the abortion process? What about software that enables crackers to break into systems (or helps US [information] war fighters break into enemy systems? or helps security teams determine if a system is vulnerable to crackers?) In many cases there is not an inherent 'proper' or `improper' usage of the project. In addition, this statement also has an implicit OR ("Assure proper goals and objectives for any project on which they work OR...") which is left unstated. Two better statements are "Inform the appropriate authorities if your project, product or work will be used for illegal purposes" , and "If you believe your work will be used for immoral (but not illegal) purposes quit, or inform your employer and ask to be moved to a different project".

A11 of the statements under Rule 3 JUDGMENT (except for the last one) are very poor. Statement 3.01 is too weak. `Professional skepticism' seems like a copout. How about "Do not let any outside, non software engineering, issues influence your professional evaluation of software or related documents you are asked to evaluate". Statements 3.02-3.06 are all redundant to 3.07 I believe 3.07 is the general and 3.02-3.06 are all specific instances so get rid of them. 3.02 is against the law and should not have to be restated so directly in a code of ethics. 3.03 and 3.09 have the 'disclosure' phrase which allows the behavior under full disclosure to all concerned parties. 3.05 and 3.06 do not have the 'disclosure' phrase yet I believe the issues are the same as those involved with 3.03 and 3.09!. Why include the disclosure phrase on 3.03 and 3.09 and not on 3.05 and 3.06? In addition, statement 3.06 introduces a paradox because if it were in the code of ethics and enforced, then (because of the self reference problem) no software engineer could participate in modifying or changing the rules of behavior for software engineers as most rules could have a financial impact to them personally and through out the industry. i.e. with this rule in place can software engineers ethically establish a certification process for themselves so they can truly become "engineers".

If it is established as a rule can they ethically change the code as any change will inevitably affect them financially and thus they will be breaking the rule?

Statement 6.07 seems out of place. It is too specific an issue and seems out of place under Rule 6: COLLEAGUES. It seems r11 stated as well. i.e. "Software Engineers shall assure co-workers know the employer's policies and procedures for protecting passwords, files and other confidential information." This is too proactive for me, am I supposed to go to my colleagues and quiz them on their knowledge .... or just inform them if the situation comes up? And why just single out those policies and procedures for "protecting passwords, files and other confidential information"? Why not include software engineering methods and documentation standards and a whole host of other things.

"Aspirational" statements. I do not have a problem with "aspirational" statements in a code of ethics. It might help to have two sections of statements under each of the rules. Those that are aspirational and those that are prescriptive. There are problems with being too specific or being too general. If you are too specific you will inevitably leave out instances of unethical behavior and there will be people that (after doing the unethical behavior) will point out that such behavior is not expressly denied. I believe that trying to cover all cases is doomed to fail as there is no way to be complete and technology changes and it is tough to keep up with it. However, if you are too general there will be people that will maintain that certain unethical behavior isn't covered under one of the broad sweeping statements or they will maintain that some ethical behavior is actually unethical because it seems to be covered by one of the statements. Basically, extremes with either thought process are no-win situations. I think the middle ground is best. Provide some aspirational statements as guidelines and some prescriptive statements as concrete examples.

Some of the statements I would leave as aspirational or change to be more prescriptive and measurable are:

1.02 "Assure that they understand fully the specifications for software on which they work" Leave as aspirational OR get rid of. The only way to measure understanding in this case is to test it which is silly. Imagine all software engineers being required to take a test on the requirements... who would grade it? Also, at what point is the test taken? Obviously one must have spent some time studying the requirements and performing analysis and design functions as software engineering is an iterative process.

1.12 "Assure ethical, economic, cultural, legal and environmental issues are properly identified, defined, and addressed" Leave as aspirational. Perhaps these issues can be codified into a software engineering method in the future some time but for now there is not enough meat on these issues to even devise a true engineering scheme for them. In addition, what is `proper' concerning these issues is not black and white. No group of people could agree on many specific examples related to these issues. A simple question like "Will producing and marketing system x change the culture of the people that use it?" does not have a definitive answer. This is not even the important question (unless you are one of the `true believers' that think that all change is inherently bad). The more important question, "Will the change be for the better of for the worse?", has no definitive answer either. Also, what is `bad' in the short term may be `good' in the long term.

1.19 "Avoid fads, departing from standard practices only when justified" Leave as aspirational OR get rid of OR restate by defining "fads" better and the conditions better. One persons `fad' is another person's silver bullet! If we don't try new things we will never progress and codifying the conditions under which new things can be tried. Codifying existing practices as the `correct' ones may reduce the likelihood that the new things will ever become standards! If I were to restate this I would include a risk analysis assessment and a `disclosure' phrase similar to those used in 3.03 and 3.09.

5.09 "Help develop an organizational environment favorable to acting ethically" This could be changed to be more prescriptive. "Help.... by acting ethically in all ways yourself, promoting this code of ethics where possible, denouncing unethical behavior whenever you see it, .. These additions (and others) may not be that statement much more `testable' but at least they give an indication of some of the practices that are expected.

7.01-7.06 "Improve their knowledge/ability/understanding... All of these are aspirational. There are many ways to make these statements more testable but I would argue against that. For example, the 'teaching profession' and many others require improvement efforts (i.e. additional course work) on a regular basis. These requirements include additional training or `classes' of training that are needed periodically. A problem with this approach is that a whole cottage industry has grown up providing some of the other professions with the `required' self improvements needed to maintain their certification. I would argue that codifying the improvement would inevitably mean establishing a minimum to which many individuals would now 'aspire' to. Also the inevitable cottage industry would arise which would be oriented towards satisfying the letter of the requirement and not the spirit.

Student 19

General Comments On This Code Of Ethics

Overall, I think it's a fairly well-conceived code of ethics, containing both strong and weak points. legitimate engineering discipline, I believe such a code must be established. In particular, its strengths lie interest, professional judgment, client and employer relationships, professional advancement, treatment of colleagues, and self endeavors. I feel the code comes up short in the are, me, though, since it seems so much harder to define a software product than it does to define hardware-oriented engineering products (electrical, mechanical, etc.). If it's intrinsically more difficult to define the product of a process, it seems only logical that it is equally as difficult to apply professional criteria to such products.

Imperatives That Should Not Belong To This Code

I feel all the imperatives have a place in the code. They all address areas of concern to the discipline of software engineering. I looked very hard at all the imperatives and for each one was able to visualize a scenario in which the imperative could have a professional impact, either on an individual or corporeal basis.

Applicability Of Any Code Of Ethics

As I stated earlier, I believe a code of ethics is imperative for any body of professionals seeking public recognition as an engineering discipline. This gives the body a common ground against which all members may be judged.

Aspirational Imperatives

I feel the following three imperatives are aspirational at best. They simply have no hard and fast measurements against which they may be gauged.

1.07 Assure proper estimates of cost, schedule, personnel, and outcome on any project on which they work. It is rare that a software project of any respectable size is delivered on time and within budget. Until a consistently reliable time/cost estimation methodology is developed for software, this imperative will continue to be an immeasurable one.

1.05 Assure proper development methodology on any project on which they work. Who defines what is the "proper" development methodology for a particular project? Many development methodologies exist, and in many cases, more than one may be applicable. Yet the question remains: how can one methodology be gauged as more proper than another?

1.06 Assure proper management on any project on which they work, including proper procedures for control of quality and risk. The same type of argument applies here. Who can determine whether or not a proper style of management is being applied? Obviously, is a project is failing, you might want to take a hard look at management. But whether or not management is proper is really only determinable by whether or not the project is ultimately successful. Therefore it cannot be fully measured as long as it is in progress.

Student 15

1. a. In general, I think this code of ethics is fairly good. It focuses on seven general areas, all of which 7 think are appropriate, and cover the ground a professional code of ethics should.

b. The following subsidiary clauses are inappropriate for inclusion:
1.19: This particular clause may encourage software engineers not to try new things that may advance the state of technology. Just because something is considered a fad doesn't mean that there isn't justification for using it. It could be argued that, several years ago, object oriented was a fad. Now it is standard practice, and that may not have happened if its use had been discouraged because it was a "fad."

5.07: Who's to say what a "salary appropriate to professional qualifications" is. Putting an artificial ceiling on what people can earn simply isn't the American way. If we let salaries be determined by the free market, people will be paid what they are worth. Amore appropriate wording may be, "Do not overstate qualifications for the sake of getting a higher salary."

5.11 -- A person's decision whether or not to participate in "civic affairs" is entirely their own, and has no bearing whatsoever on their status as an upstanding professional software engineer. This clause should be =v that we should need codes of ethics in it is necessary. It is important to have a minimal at *which to* hold people, because, as with any profession, a small number of people lacking ethics will find their way in.

2. The following clauses are not easily measurable, and should be at least considered for rewording:

5.01: It is impossible to measure what is or is not a "reputable" business. Certainly, some businesses (i.e. the Mafia) are considered disreputable by all and should not be associated with by right-standing engineers. Other businesses fall into a gray area, and it is simply not possible to threshold of "reputability."

1.05: How does one measure whether or not a "proper" development methodology has been used? Perhaps a better wording would be, "Upon choosing a development methodology, the software engineer should document and be able to defend the reasoning behind choosing that particular methodology." This wording would at least allow the question to be answered yes or no. 1.09: This clause is worded very similarly to 1.05,

and is not measurable. I would suggest, "A software engineer should be capable of properly defending his documented goals and objectives for any projects." 3.01: What is "professional skepticism" and how is it measured? I assume this means that an evaluator should always assume that the product will "break" until it is proven otherwise. If this is the case, the clause will probably remain aspirational, but should be worded more clearly. 5.09: It is impossible to measure whether or not one has contributed to "developing an environment favorable to acting ethically." This clause, while necessary, is probably inherently immeasurable. 5.09: At what point does promoting ones own interest come at the expense of the profession? Once again, there is no way to establish an acceptable threshold. This clause is also inherently immeasurable.