Demons in Software Development

Richard G. Epstein
Department of Computer Science
West Chester University of Pennsylvania
West Chester, PA 19383

Introduction

During the spring of 1998 the author was asked to give a plenary session luncheon talk at the ACM flagship conference for that year, the ACM Policy 98 Conference. He was asked to expand and elaborate upon the ideas that were found in his essay, "The Wheel," [1] that had just been published by the ACM SIGCAS publication, Computers and Society. The talk that the author gave on that occasion was perhaps a bit unusual for an ACM conference. The author was trying to make the point that the inner being of the computer professional influences the quality of his work on many different levels, including the social and ethical implications of the products that he produces.

The term "inner being" refers to the psychological reality of the computer professional. "Inner being" refers to both afflictive and destructive states of mind as well as compassionate and caring states of mind (for example, see [2,3]). These psychological realities, in turn, have an impact on the software products that computer professionals create, including the social and ethical implications of these technologies.

An article that had a great impact upon the author's thinking along these lines was "Beyond Blaming" by Jean McLendon and Gerald Weinberg [4]. Jean McLendon is a psychologist and software industry consultant and Gerald Weinberg is the well-known software engineer who wrote the influential book The Psychology of Computer Programming [5]. In their capacity as industry consultants McLendon and Weinberg have seen how the inner being of software developers is a key factor in the success and failure of software projects. "Beyond Blaming" introduces the concept of "congruence," which McLendon and Weinberg define as the alignment between the inner and outer realities of an individual who is involved on a software project (whether a developer or a manager). "Congruence" applies both to the individual developer or manager as well as to the environment in which she works, the work culture. Thus, in a congruent work culture, what an individual says and does is consistent with what that developer thinks and feels. By contrast, in an incongruent work culture, expressing the truth that one thinks and feels is a risky proposition. An excellent description of what incongruence can do to a software project is found in Effy Oz's account of the Confirm scenario [6]. Although Oz does not use the term "incongruence," the incongruence at AMRIS (the developer organization in the scenario) is obvious in Oz's account.

An important point that McLendon and Weinberg make in "Beyond Blaming" is that when the work culture is incongruent, this greatly decreases the likelihood that a software project will succeed. In contrast, a congruent work culture can be a critical factor for the success of a software project.

McLendon and Weinberg concentrate on one specific symptom, blaming, that characterizes an incongruent work culture. "Beyond Blaming" gives an in-depth analysis of the implications of blaming for a software project. In this paper we shall characterize "blaming" as just one of many workplace demons (afflictive states of mind) that might impair the success of a

software project. But first, let's examine in more detail how McLendon and Weinberg describe the blaming demon in "Beyond Blaming."

McLendon and Weinberg make a startling claim. They assert that "[b]laming is the dark secret underlying the failure of many projects." They then go on to list major ways in which a blaming culture hurts a software project. These include (1) causing people to commit to unreasonable plans (as was the case in the Confirm project), (2) causing people to hide facts that managers need to control the project, (3) stifling creativity, because a creative idea that does not work will cause the person who contributed that idea to be blamed, (4) causing people to spend a lot of time protecting themselves in anticipation of a "day of reckoning," (5) forcing people who actually can focus on their work to waste a lot of time trying to verify the truthfulness and reliability of communications that they receive, and (6) greatly decreasing developer work satisfaction and productivity. ([4], pp. 36-7)

In contrast, McLendon and Weinberg characterize congruence as "the bright secret underlying the success of many projects." A congruent work culture promotes project success in a variety of ways. These include (1) people commit to plans only after open discussion and negotiation, so that the plans tend to be realistic, (2) managers get information they need so that they can focus on solving problems that arise, (3) people willingly contribute their creative ideas to solve problems, (4) people focus on getting their jobs done and helping others to get their jobs done, (5) people spend an appropriate amount of time (i.e., not an inordinate amount of time) verifying communications, and (6) people enjoy high work satisfaction and high productivity. ([4], p. 39)

In this paper, "blaming" is viewed as just one of dozens of demons that can influence the success or failure of a software project. In the author's software engineering course, he uses a class exercise (or, what he calls a "classercise") that is intended to make students aware of these demons. Some of these students have years of experience in the software development, and these are the students who react most strongly to this classercise. Sometimes the reaction is very strong, because these students have encountered these demons on various software projects and these demons made their workplace experiences very negative.

The remaining sections of this paper will discuss: (1) The nature of demons and the role that they play within our human reality, (2) Forty-three demons that can infect the software development workplace, (3) How demons can impact a software project, (4) Remedies for demons in the software development workplace, (5) Antidotes, or positive states of mind, and (6) Demons and professionalism.

The Nature of Demons / The Role They Play within the Human Reality

A demon can be characterized as an afflictive state of mind; in other words, a state of mind that causes suffering. One way of understanding the role that demons play within the human reality is to say that all excessive human suffering is the result of demons. Of course, it is difficult to explain exactly what is meant by "excessive human suffering." For example, if someone is in great pain due to an illness or an injury, then there is suffering involved. However, demons are psychological realities that take these natural responses and exaggerate them, causing suffering that is not necessary. Demons can cause suffering even when there is no objective reality that would seem to be the cause of the suffering. We might be healthy, have a

roof over our heads, have adequate food, and all of that, but we can still be afflicted by legions of demons. These legions of demons are a basic human reality.

The author has been using his "Demons in the Software Development Workplace" classercise in his software engineering class for at least five years. However, this past summer (2003) he read a Zen story that gave him a new insight into the nature of demons and how the demon problem might be understood and addressed in the work environment. This story is found in Ezra Bayda's book, Being Zen [7]. Here is that Zen story from Bayda's book, quoted in complete detail:

> "An ambitious long-time meditator came to see a Zen teacher. As soon as the student sits down, the teacher asks, 'What's the basic human problem?' The student ponders this, then answers, 'We're not awake.' The teacher says, 'Yes, but those are just words. You're just thinking.' And ringing the bell, he sends the student away.
>
> Perturbed, the student continues to ponder, 'What is the basic human problem?,' determined to figure it out. A week later he returns. The teacher says, 'Well, have you figured out what the basic human problem is?' The student replies, 'Yes. The basic human problem is that we think too much. We're identified with our thinking. We believe our thoughts.' The teacher answers, 'Again, you're just thinking. You have to see the basic human problem in yourself.' The student leaves feeling very dejected.
>
> Wanting to find the right answer, the student pulls out all his Zen books to read and study. When he returns to see the teacher, he's almost strutting, he's so sure he knows the answer to this question. Seeing the state he's in, the teacher asks, 'What's the basic human problem?' And the student says, 'There is no problem!' He's so happy with his answer. The teacher just stares at him and says, 'Then what are you doing here?' In that moment the student instantly deflates. His shoulders drop; his head drops; he feels totally humiliated. Peering at him, the teacher asks, 'What are you experiencing right now?' The student, without even looking up, says, 'I just feel like crawling into a hole.' At this point the teacher says to him, 'If you can fully experience this feeling, then you'll understand the basic human problem.' ([7], pp. 47-8)

What's going on here? The Zen teacher is trying to get the student to become aware, deeply aware, of his own dominant demon. That demon is a consequence of a basic life decision that the student has made (using Bayda's terminology). The student has made the basic life decision that he must be the best, that he must shine in every situation. The Zen teacher sees this in the student, so the teacher tries to provoke this demon within the student. Once the demon has been provoked, the teacher then has the opportunity to help the student to become aware of the reality of that particular demon and to see how that demon has caused so much suffering in the student's life. Once the student comes to understand this particular demon (whether it is called Arrogance or Perfectionism or Personal Glory), the student then has the opportunity to understand the other demons that might be causing him to suffer.

The Zen teacher can be viewed as a metaphor for the Universe and the student is a metaphor for each one of us. One can say that the Universe is constantly provoking our demons, but we rarely have the kind of awareness that it takes for us to see precisely what is going on, to see the impact that the demons have upon us and upon the people in our environment. This kind of awareness (or, mindfulness) is necessary in order to transform the demons into positive energies that can make us more peaceful, more open to change, more open to the challenges that life inevitably presents, and more productive in our work environments.

Ezra Bayda says a few things about this story that will help us to understand the role that the basic human problem plays in our life:

> "The essence of the basic human problem is that we live a substitute life. From our basic human need for protection, security, and comfort, we've fabricated a whole maze of constructs and strategies to avoid being with life as it is. And as a consequence of believing in this substitute life we are disconnected from awareness of our true nature, our naturally open heart." ([7], p. 48)

In this story the student's substitute life has to do with his firm conviction that he must be the best, that he must shine in every situation. This kind of basic life decision is often the result of the individual's attempting to avoid feelings of inadequacy. The demons that we shall discuss are manifestations of a basic lack of faith in our essential human nature. Quoting again from Bayda:

> "Perhaps we feel that there's some hole inside of us that needs filling. Perhaps we even feel the terror of utter helplessness or of being totally alone. When we feel this anxious quiver in our being, our natural instinct for protection kicks in. And from this natural desire for security and comfort, we begin to fill in that hole and cover over that core of pain." ([7], pp. 48-9)

Thus, a demon is a protective mechanism within the human psyche. This mechanism is a pattern of energy that includes thoughts, feelings, behaviors and spoken words (or, speech acts). A demon arises when we feel that there is some kind of flaw or imperfection at the core of our being, and this mistaken belief causes the ego (psyche) to generate all sorts of energies (thoughts, feelings, behaviors, speech acts) in its desperate attempt to protect itself.

In "Beyond Blaming," McLendon and Weinberg focus on just one demon, the blaming demon, but they make the point that this demon often has its origins in a person's lack of self-confidence. This lack of self-confidence is the hole at the center of the person, and the blaming demon is the energy, the feelings, the thoughts, the behaviors, and the spoken words that protect that person from discovering that hole.

McLendon and Weinberg list a series of six steps that an individual software developer must go through in order to vanquish the blaming demon and eventually to make a positive contribution to his or her software development environment. These six steps, recast below in terms of the general demon concept, apply to all of the demons that we shall introduce in the next section of this paper, not just the blaming demon:

1. Awareness: Become aware of the role the demon is playing in one's own psyche and in the work culture. See the impact of that demon in the workplace.
2. Acceptance: Accept the role that one plays in nurturing this demon. Forgive oneself and forgive one's colleagues, because demons are a basic human reality. They are, in fact, manifestations of the basic human problem.
3. Authorship: Realize that one has the freedom to control, and possible even to eradicate, the demon from one's psyche and from one's work culture.
4. Articulation: Make appropriate public comments on the presence of the demon, whether in oneself or in the work culture. For example, if a personal demon has manifested and has interfered with the success of a project, one might have to ask one's colleagues for forgiveness. If a particular demon (e.g., Anger, Resistance to Change) exists in the work culture and if that demon is having a disruptive impact on the software project, one might have to bring that demon to the group's attention.

5. Application: Apply what one has learned about a particular demon to make constructive changes in the workplace, helping the team to heal.
6. Activism: Help to create an open and creative workplace, where demons are transformed into healing and constructive energies.

The blaming demon in an incongruent work culture infects just about everyone. The demons that are listed in the next section are less likely to infect an entire organization, although they might. These demons are more individualistic, more personal. Yet, the reality is that these individual demons have the power to feed on one another. Unless there is an awareness in an organization about how to handle the demons in an effective manner, there is likely to be an explosion of negative and destructive energies. Incongruence is not a necessary pre-condition for the existence of these and other demons, although incongruence might be a factor in their proliferation.

Forty-Three Workplace Demons

In this section we shall list forty-three demons that can infect the software development workplace. Each demon is described by means of a quote that represents a typical thought pattern or speech act that this demon might create in its infected host. Again, a demon is viewed as an energy pattern. This energy pattern includes feelings, thoughts, behaviors, and speech acts. Of course, there are many more demons than those listed here and the reader is welcome to make up his or her own list of relevant demons.

Here are the forty-three demons in the author s list along with some typical thought patterns and / or speech acts associated with each:

1. Personal Glory: I do everything for my own personal honor and glory. I want people to praise me and the more praise the better. My goal is to be famous, so that everyone will know my name and sing my praises. "
2. Impatience: I can t stand it when people waste my time. I take every red light and stop sign as a personal affront. When people say in ten sentences what could have been said in just one or two, I really fume. "
3. Anger: I have a short fuse. Sometimes, when things aren t going my way, I use anger to manipulate people. That seems like the best way to get people to take me seriously. Also, once someone has seen me blow my stack, they are less likely to cross me up in the future. "
4. Intolerance: Obviously, I belong to the best ethnic group that ever was. I have a cartoon-like stereotype for every other ethnic group that I take to be the truth. Then, when someone from another ethnic group fits my stereotype, I can feel superior to that person. "
5. Sexism: Women belong at home, cleaning and making babies. I don t see what women are doing in the computer field. They just don t have the smarts of us guys. "(Of course, sexism could go in the other direction.)
6. Stubbornness: Once I set my mind on the correct way of doing something, I will stick to that idea to the bitter end. Some people say that I am stubborn, but you can t accomplish anything without perseverance. "
7. Laziness: Only a damn fool will put forth more effort than is required. My objective is to do as little work as possible and to have a good time. "
8. Workaholism: Work is my god. I work seven days a week, at least twelve hours a day. This gives me a sense of personal worth. I don t see any need to diversify, to branch out, to experience new things. Those kinds of diversions will just short circuit my career.

Furthermore, if I stop for more than a few minutes, I might have to take a good look at myself. "

9. Greed: "I love money and possessions. When I code I see a new gigantic SUV in my code, and a new house by the ocean. My main goal in life is to acquire as many possessions as I can. So what if my lifestyle is destroying the planet? All that matters is that I have lots of possessions so that I have a sense of being a success, of being a somebody. Whoever ends up with the most toys, wins. "
10. Alcoholism and Intoxicants: "What I do with my free time is my problem. It's none of your business. I can spend a whole night drinking and be just fine the next day. It doesn't affect my work, especially since my wife left me. And if I beat on my kids, they were really getting out of hand. Now, I have even more time to get bombed. Where did I put my car keys? "
11. Individualism: "This teamwork stuff is just a bunch of nonsense. I work for me and for me alone. It's important that this project has my personal stamp on it. If the project doesn't say ME, why work on it? Working with other people just brings up a lot of feelings I'd rather not have to deal with. "
12. Fear / Anxiety / Worry: "The future is pregnant with disastrous possibilities. Maybe I will get downsized. What if this project is not a success? What if Sam gets the promotion that I've been trying to get? What if my stock market investments go down the toilet? What if I run out of 'What if?' questions to ask myself? "
13. Dishonesty: "Insofar as I am concerned, lying is not a problem, so long as you can get away with it. My boss asked me whether I would finish my project on time, and I told him, 'Yeah, sure.' Later, I'll think of a good excuse for why the project isn't finished. By then, I can have a new job lined up. After all, I don't know half the things I put on my resume. "
14. Gossip: "I think it is important to talk about other people as much as possible. I mean, by spreading rumors and gossip, I am performing a public service. So what if I embellish on the truth? The important thing is that I hold people's interest. I'm just like CNN, but on a smaller scale. "
15. Backbiting: "By putting other people down, I affirm my own natural superiority, which should be obvious, or haven't you noticed? "
16. Negativity: "Whatever the situation, I know how to put a negative spin on it. That's just the way my mind works. If Thomas Edison had been like me, you'd be reading this by candlelight. "
17. Selfishness: "The easiest way to lose your manhood is to care about someone else. No way! I look out for numero uno! "
18. Perfectionism: "Whatever I do must be absolutely perfect, because if it's not perfect, it will just open me up to criticism, which would open up all sorts of unpleasant feelings. I cannot live with myself if my work is not absolutely perfect. "
19. Sloppiness: "I just don't have the awareness to be neat or orderly. All great geniuses were disorganized and sloppy. So what if I don't keep careful track of my work? I know where to find things in that pile on my desk. "
20. Carelessness / Mindlessness: "I just don't have the patience to attend to details. If I leave out a semicolon in my code, the compiler will find it. I'm more interested in the bleeding edge than the little details that any moron could attend to. "
21. Blaming: "Insofar as I can tell, I have never been at fault. It's always someone else who screws up, and I usually can spot that person a mile away. "

22. Inability to Listen: "I doubt very much that anyone can tell me anything that I do not already know. The trouble with listening is that listening might cause me to change my mind, which is more trouble than it's worth."
23. Inability to Speak to Power: "I find it difficult to be honest with my superiors, because to be honest is to risk having the wrath of God fall down on your head. Actually, you may not realize this, but I am the strong, silent type."
24. Resentment: "I love to cast situations in such a way that I see that other person as doing me personal harm. What fun! This allows me to build up a mental construction of the situation that keeps me in a state of anger. That's a lot better than having to talk things through or trying to understand that other person."
25. Jealousy: "I can't stand it when someone else does a good job, because it makes me feel insecure and worthless. When the boss praises someone else on the team, I seethe with resentment. If a colleague gets an 80 gigabyte PC on her desk and I only have 40 gigabytes, there will be hell to pay!"
26. Vengeance: "If someone harms me, you can be damn sure that I am going to get even. After all, I am perfect. Also, I know that the other person deliberately caused me harm, because my construction of reality is infallible. Do you doubt my infallible construction of reality? Jerk!"
27. Lack of Respect for the Other: "In any relationship, the other is hardly worth considering. I am sure that the other has nothing in common with me because my construction of reality is the only true construction of reality, and in my construction of reality, I am numero uno."
28. Insensitivity: "I think it is okay to hurt the feelings of another person, so long as (1) it gives everyone a good laugh, and / or (2) it makes me feel superior to that other person."
29. Indecisiveness: "I can't stand making a decision. I love it when someone else makes my decisions for me. Then, if things go wrong, he can take the blame."
30. Absolutism: "Everything in this universe is black or white. I know what is good and I know what is evil. I am privy to know who is good and who is bad, who is right and who is wrong. What an accomplishment!"
31. Avoidance: "I will do everything in my power to avoid discomfort. I will do everything in my power to avoid confrontation, to avoid unpleasant situations. I think in a previous life I must have been a hermit living in a cave. Ah, for the good old days."
32. Intellectual Superiority: "All that matters is that I establish my intellectual superiority over everyone in my environment. If I can do that with an incredibly subtle and incomprehensible solution to a programming problem, so much the better. The important thing is for you to realize that I am smart and that you are dumb."
33. Non-conformity: "I am not the sort of person who goes along with the herd. I think all this business about a well-defined software process is just an attempt to get everyone to conform, and if I have to conform, I lose my individuality, my sense of who I am."
34. Self-loathing: "Don't bother trying to cheer me up. That defect you found in my code proves that I am worthless. In some perverse sense, I enjoy my worthlessness. It gives me a great excuse for not taking any risks."
35. Comfort: "I know my comfort zone, and I will do everything in my power to protect it. If you need more details, my comfort zone is six feet by six feet with a seven foot ceiling. There's a little cot and a potty."

36. Technology: "There isn't a problem known to humanity that cannot be solved by throwing technology at it. Whatever the problem, there's got to be a new tool that will solve it. If this project is going to hell, it's because we don't have the latest CASE tool."
37. Arrogance: "I represent the truth, the whole truth, and nothing but the truth. You got a problem with that? My opinions are always right because they are mine. Your opinions are always wrong, because they are yours. Am I making myself clear?"
38. Power: "The most important thing in the world is power. I love to have power over people, to hold their fate in my hands, to be worshipped and adored. Now that we understand one another—do as I say."
39. Skepticism: "It'll be a cold day in hell before you can pull one over on me. My intelligence is so sharp that it bleeds. I can look at any proposed solution to a problem and I can see its inherent flaws. Insofar as I can tell, all solutions are flawed. Just let me know if I can be of help on this project of yours."
40. Pride: "I am never wrong. I am never wrong. I am never wrong. Now, acknowledging that I am never wrong, if you can still find a reason why I should say, 'There's a defect in my program,' then maybe you should go and have your head examined."
41. Boredom: "Bore-ing! Life is intrinsically boring. That's just the nature of reality. So, now are you going to tell me to see the awesome beauty in that flower blooming on the side of the road? What are you, some kind of hippie?"
42. Security: "The bottom line in every decision that I make is security. My position is secure and it will remain secure just so long as I remember not to rock the boat."
43. Resistance to Change: "Look, this is the way things have been done around here for many years. You whippersnappers need to understand that I was programming computers before some of you were even born. The way we've been doing things in the past is the way we had better do them in the future."

How Demons Can Impact a Software Project

The "classercise" that the author mentioned at the beginning of this article can evolve in a variety of ways. One approach is to have the students break up into teams. The students on each team pretend that they are meeting to discuss a software problem that has arisen on a project. Their goal is to come up with a solution. The author has a deck of forty-three "demon cards." Each demon card has the name of one demon on it. Each student picks one card and plays the role of a software developer infected with that demon. The idea is to explore the implications of that demon for the team's ability to work towards a solution for the problem in a timely manner.

The students are given the following questions to consider:

1. How can an individual team member, infected with this demon, interfere with her team's efforts to develop a quality software product?
2. How can this demon harm a software project if it infects many team members, not just one?
3. How can this specific demon cause other demons to manifest within the team? What are the implications of this "proliferation" of demons?
4. A demon often makes demands. What kinds of demands might this demon make upon a software developer? For example, the workaholism demon might make tremendous demands upon a developer's time and it might also warn him against activities that are contrary to the demon's values. The workaholism demon might

command that the developer work twelve hours a day, seven days a week, and it may warn him against taking a day off because doing so might harm his career.

5. Is this demon masking a positive energy (attitude) that has gone out of balance? For example, stubbornness can be viewed as a form of persistence that has been taken to the extreme. Hard work is a virtue, but workaholism is hard work taken to the extreme. The positive qualities become demons when they become exaggerated, inappropriate, or extreme.
6. Are there positive qualities that can serve as antidotes to this particular demon? For example, forgiveness is a good antidote for Resentment and openness is a good antidote for Inflexibility. We shall discuss these positive qualities (or, antidotes) later in this paper.

Each demon has its own implications both for the individual developer and for the software development team. Some of these demons might just infect one particular developer. That demon might have an impact upon the ability of that developer to make professionally sound decisions about his work. The manifestation of that demon might have an impact upon other members of the team. The other team members might find themselves making adjustments in order to deal with the demon that has manifested. It would not be unusual for the other team members to respond with their own demons. For example, if one developer is infected with the Lack of Respect for the Other demon, then that developer's teammates might find themselves responding with the Vengeance, Anger, Avoidance, or Resentment demons. The result can be an explosion of demons, like in a nuclear chain reaction.

Remedies for Demons in Software Development

There would seem to be no simple solution to the problem of demons in the software development workplace. If the eradication of demons from the human reality were a simple matter, we wouldn't be living in a world with so much violence and hatred and injustice. So, we should not expect the eradication of demons from the workplace to be a simple undertaking. We are, after all, dealing with the basic human problem. Awareness would seem to be the key to addressing the problem of demons in the software development workplace. Let us look at some things that individual developers and managers can do to minimize the impact of demons when they do arise.

Individual developers. There are two dimensions of individual responsibility with respect to demons in the workplace. First, the individual developer must take responsibility for his or her own demons. Second, the individual developer must understand how best to respond when demons manifest within other developers.

The individual developer must take responsibility for his or her own demons. If every developer developed a deep awareness of her or her own reality, then this problem would be solved. However, this is a difficult task. Nonetheless, each individual developer must recognize the demons within himself or herself and must recognize the negative impact that those demons can have on that individual's career and professional reputation. Each demon has its own consequences in terms of the quality of one's work, the impact that the demon has one one's team, and the overall success of the project. The guidance provided by McLendon and Weinberg is relevant here, in that the individual needs to go through at least the initial steps of awareness, acceptance and authorship in order to transform the negative energies into positive ones. The

key is to discover positive energy patterns, new ways of reacting, new patterns of thinking, of behaving, and of speaking. Earlier, we referred to these positive energies as "antidotes." We shall explore antidotes in the next section.

The individual developer must also come to recognize the demons that arise within his or her colleagues. It is important to be objective and not judgmental about those demons. Here the steps are to recognize the objective existence of a demon in a colleague, accept that as part of the reality of that person, and author compassionate methods for helping that person to transform himself or herself, both for that person's own benefit and for the benefit of the entire team. This is not an easy task, but one that requires quite a bit of thought and reflection.

The great danger is that demon A in one software developer might have the power to elicit demons B, C, D, and E in another developer. In a team context, one might face an explosion of demons that prevents the team from focusing on solving the technical problems that need to be solved, objectively, effectively, and in a timely manner. Therefore, before one can help another developer with his or her demons, one must really be aware of one's own demons and the kinds of circumstances that stimulate those demons. The manifestation and interaction of demons in the workplace is a very complicated process.

Management responsibility. Management must become aware of the work culture and the ways in which the work culture might stimulate and promote the manifestation of demons. For example, if the work culture makes unreasonable demands upon developers, this might encourage a host of demons, including Self-Loathing, Arrogance, Jealousy, Vengeance, Insensitivity, Avoidance, Lack of Respect for the Other, and so forth.

One can assert that every management policy or decision has the potential to elicit demons. One cannot necessarily state that a policy is poor just because it elicits demons. For example, a responsible decision to pursue software process improvement within an organization might elicit many demons within the developers (Resentment, Inflexibility, Arrogance, Non-Conformity, Individualism). The key challenge for management is to overcome these demons with wise management practices. Management must work to inspire the positive qualities (e.g., openness, flexibility, passion for learning, passion for excellence, team-orientation, humility) that serve as antidotes for the workplace demons.

Bill Pitterman documents how Telcordia reached CMM Level 5 [8]. An important factor that helped to prevent the proliferation of workplace demons at Telcordia as that company strove to achieve a high level of maturity was management's commitment to giving developers a feeling of "ownership" in terms of the software process that was being developed. Another factor was Telcordia's commitment to "common sense" as opposed to "mindless bureaucracy." The very manner in which an organization pursues process maturity will either encourage or discourage the manifestation of workplace demons.

Another way of characterizing a healthy software development environment is to draw upon Weinberg's concept of "egoless programming" [6]. An environment where egoless programming is practiced is clearly less conducive to creating demons than an environment in which individualism reigns and developers are chastised and humiliated for every defect they inject into the code.

Management might use training seminars and discussion groups to teach developers about workplace demons. One aspect of such a training program might be to bring out the humorous side of demons in the workplace. This kind of self-deprecating humor, when used in the correct manner, can help developers (and managers) to see themselves in perspective, to see

their human foibles and their basic humanity. In other words, humor can be a powerful tool for helping a software development team to see its weak points and to evolve beyond them.

Of course, there will be software developers who will refuse to change and who are so governed by fear that they will refuse to confront and transform their demons. Steve McConnell in his paper, "Problem Programmers," [9] explores this reality. Although Steve McConnell does not use the terminology of demons, his description of the problem programmer is a description of a person who has many of the demons that we have described above. Here is part of McConnell's description of some of the characteristics of what he calls a "problem programmer." The author has expanded this quote from McConnell's article to include the names of some of the demons that might be lurking behind the counter-productive behaviors that problem programmers often exhibit:

> "Low productivity by itself usually isn't the only problem. Strained to the limits of their abilities by the coding activity itself, low productivity programmers are either unable or unwilling to follow project coding conventions or design standards [Lack of Respect for the Other, Absolutism, Intelligence, Arrogance, Non-Conformity, Resistance to Change]. They remove few or no defects from their code before they integrate it with other people's work, or before other people are affected by it [Laziness, Sloppiness, Individualism, Stubbornness, Personal Glory]. They can't estimate their work reliably because they don't know for sure whether they will even finish [Inability to Speak to Power, Inability to Listen, Indecisiveness, Avoidance, Vengeance, Non-Conformity, Pride, Dishonesty, Security]. Considering the absence of direct contributions to the project and the extra work created for the rest of the team [Insensitivity, Lack of Respect for the Other, Avoidance, Selfishness], it's no exaggeration to classify these programmers as negative productivity programmers." ([9], pp. 128-7)

McConnell goes on to state that about ten percent of professional programmers are negative productivity programmers. He strongly recommends that negative productivity programmers be fired. The emphasis in this paper upon creating a workplace environment in which there is an awareness of the demons and a concerted effort to remove them does not contradict what McConnell is saying. There are cases in which a developer may be so deeply afflicted with his or her personal demons that the person needs to be removed from the development team. Otherwise, the success and the well-being of that problem programmer's colleagues will be placed at risk.

Antidotes, or Positive States of Mind

A fundamental goal is not only to become aware of the demons and to author creative and constructive responses to the demons, but also to transform the demonic energies into positive energies. We call these positive energies "antidotes." The antidotes are specific energy patterns (that is, feelings, thoughts, behaviors, and speech acts) that act against one or more specific demons. Just as McLendon and Weinberg stress that a congruent work culture is the cure for a blaming culture, so an environment that is rich with the healing power of the antidotes will mitigate against the development of the demons.

Table 1 shows most of the demons that we had in our original list paired with the corresponding healing and constructive energies that we are calling antidotes. The antidotes are the medicines that the individual developers and managers in the software development workplace must apply in order to prevent the demons from spreading and causing damage to the

work culture. The one antidote that Table 1 does not list explicitly is "love." This is because love would seem to be an antidote for just about all of the demons that we have listed.

Table 1. Demons and the corresponding antidotes

Demon:	Antidote(s):
Personal Glory	Self-confidence, appreciation for oneself and for the other, objectivity, humility
Impatience	Patience, mindfulness
Anger	Level-headedness, mindfulness, compassion, forgiveness, objectivity
Intolerance / Sexism	Tolerance, embracing the other with respect and compassion, objectivity
Stubbornness / Resistance to change	Flexibility, ability to listen, openness to the challenges that life offers, faith
Laziness / Workaholism	Balance, passion about one's work, seeing the value of one's work while seeing one's work in perspective, embracing life in its diversity and in its totality
Individualism / Selfishness / Greed	Team-orientation, egoless programming, caring about and helping others, generosity, objectivity
Fear / Anxiety / Worry / Negatvity	Faith, confidence, objectivity, enthusiasm, the ability to create positive "what if" scenarios as well as negative "what if" scenarios
Dishonesty	Truthfulness, respect for the other'
Gossip / Backbiting	Responsible speech, ethical speech, mindfulness, care and respect for the other
Perfectionism / Sloppiness / Carelessness & Mindlessness / Absolutism	Mindfulness, ability to forgive oneself and others, care about one's work, openness, flexibility, openness
Alcoholism and Intoxicants	Sobriety, professional responsibility, mindfulness, respect for self and the other'
Blaming	Generosity, seeing and acknowledging the positive contributions of others, objectivity, truthfulness
Inability to Listen	Ability to listen, mindfulness, openness to the challenges that life offers, attitude of the student
Inability to Speak to Power	Truthfulness, integrity, inner power, honesty, courage
Resentment / Jealousy / Vengeance	Objectivity, forgiveness, generosity, compassion
Lack of Respect for the Other	Respect for self and other, objectivity, generosity
Insensitivity	Mindfulness, sensitivity, compassion, caring
Indecisiveness / Avoidance / Comfort	Objectivity, courage, confidence, truthfulness, faith, flexibility
Intellectual superiority / Pride	Wisdom, objectivity, forgiveness of self, respect for others, humility
Non-conformity	Self-confidence, humility, respect for self and the other, mindfulness
Self-Loathing	Forgiveness, objectivity, wisdom, confidence

Technology	Objectivity, professional knowledge, respect for process and hard work, awareness of the human reality
Skepticism	Objectivity, seeing the potential within others and within ideas, faith, courage
Boredom	Mindfulness, openness, courage
Security	Objectivity, mindfulness, confidence, faith

The subject of demons is vast because we are referring to a basic human reality. However, there is a positive side to demons that the author feels just cannot be ignored. In many spiritual traditions, the antidotes are considered to be the "true nature" of the human being, or attributes of the "true self." This perception can help us to see that the ultimate purpose of demons (in the cosmic view of things) is to help sentient beings to discover the antidotes, that is, to discover their true natures, their true selves.

Demons and Professionalism

One way to eliminate demons from the software development workplace is for management to stress that software development is a profession and that software developers are expected to adhere to a code of ethics and to understand the profound implications of such a code. The Software Engineering Code of Ethics [10] states that the software engineer should "strive to be human." This is a fundamental goal of any professional. In addition, the software engineer should strive to be a good professional (in general) and a good software engineer (in particular).

Some aspects of the Software Engineering Code of Ethics can be interpreted as asserting that it is the responsibility of the software developer and / or software manager to bring certain demons under control, demons such as Dishonesty, Gossip, Backbiting, Intolerance, Sexism, Sloppiness, Inability to Listen, Inability to Speak to Power, Carelessness / Mindlessness, Jealousy, Lack of Respect for the Other, Blaming, and Non-Conformity. In other words, being a software engineering professional is not consistent with allowing these demons to dominate in the workplace.

The Software Engineering Code of Ethics is built around eight domains or principles. Each principle carries its own specific guidance for the software engineer. For example, here is a portion of the section of the Software Engineering Code of Ethics that relates to the treatment of one's colleagues (Principle 7 in the Code):

> "Software engineers shall treat all those with whom they work fairly and take positive steps to support collegial activities. In particular, software engineers shall, as appropriate:
>
> 7.01 Assist colleagues in professional development.
>
> 7.02 Review the work of other software engineers, which is not in the public domain, only with their prior knowledge, provided this is consistent with safety,
>
> 7.03 Credit fully the work of others.
>
> 7.04 Review the work of others in an objective, candid, and properly documented way.
>
> 7.05 Give a fair hearing to the opinion, concern, or complaint of a colleague." ([10, p. 116)

Principle 7 (Colleagues) contains additional precepts, but it should be clear from this excerpt that when a developer or manager is infected by a specific demon (such as Arrogance or Inability to Listen or Intellectual Superiority or Personal Glory), it would be difficult for that individual's behavior or speech to be consistent with the Code of Ethics.

A professional is expected to follow professional practices. For software developers, this would imply the need to evolve away from chaotic software processes (what are called "immature" processes in CMM terminology) towards well-defined and effective software processes. An effective software process must be rooted in common sense. Furthermore, as we mentioned earlier in commenting how Telcordia achieved CMM level 5, the software developers must have a sense of ownership when it comes to the process and when it comes to updating the process. The developers must see the software process as consistent with their experiences in terms of what constitutes "best practices." Processes that are overly burdensome, not supported by the developers, and not rooted in common sense will arouse workplace demons, including the possibility that developers might even try to sabotage the process improvement effort.

This author believes that a well-defined and appropriate software process *by itself* mitigates against the propagation of destructive demons in the workplace. (This would certainly be an interesting thesis for an empirical study.) We can go back to Ezra Bayda's comments given earlier in this paper in order to see that this is likely to be the case. Demons arise from fear. When developers do not know what the expectations are, when organizations are making plans without appropriate data, when the software process is inflexible and not based on common sense, there is likely to be more chaos and more fear in the organization. Consequently, while a well-defined and improving software process cannot solve the demon problem, it certainly can help. On the other hand, a software process that is too rigid and that is not appropriate for a given development environment is likely to create its own demons (Anger, Skepticism, Vengeance, Lack of Respect for the Other, Dishonesty, Negativity, etc.).

Conclusions

This paper explored the reality of demons in the software development workplace. The author realizes that the language used in this paper is a bit unusual, but he nonetheless believes that demons are a significant reality in software development. Organizations need to understand how and why demons arise, and they need to develop the means for preventing demons from arising and for handling demons when they do arise. One key to preventing demons from arising in the software development workplace is to pursue the goals of true professionalism. Codes of ethics and sensible software processes can play a role in mitigating against the manifestation and proliferation of demons in the software development workplace.

References

1. Epstein, R., "The Wheel," Computers and Society, Spring 1997. Also to appear in Readings in CyberEthics, 2nd edition, Spinello and Tavani (editors), Jones and Bartlett, Boston, 2004.
2. Lama, D., and Goleman, D., Destructive Emotions: How Can We Overcome Them? A Scientific Dialogue with the Dalai Lama, Bantam Books, New York, 2003.
3. Richmond, L., Work as a Spiritual Practice, Broadway Books, New York, 1999, 258 pp.
4. McLendon, J., and Weinberg, G., "Beyond Blaming: Congruence in Large Systems Development Projects," IEEE Software, July 1996, pp. 33-42.

5. Weinberg, G., The Psychology of Computer Programming, Dorset House, New York, 1998.
6. Oz, E., "When Professional Standards are Lax: The Confirm Failure and Its Lessons," Communications of the ACM, October 1994, 37(10), pp. 29-36.
7. Bayda, E., Being Zen, Shambhala, Boston, 2002.
8. Pitterman, B., "Telcordia Technologies: The Journey to High Maturity," IEEE Software, July/August 2000, pp. 89-96.
9.McConnell, S., "Problem Programmers," IEEE Software, March/April 1998, pp. 126, 127-8.
10. Gotterbarn, D., Miller, K., and Rogerson, S., "Software Engineering Code of Ethics," Communications of the ACM, September 1997, 40(11), pp. 110-116.

About the Author

Dr. Richard G. Epstein is a Professor of Computer Science at West Chester University of Pennsylvania in West Chester, PA. His main interests are software engineering, computer ethics, and the social implications of computing. These interests are reflected in his book, The Case of the Killer Robot, and in many of his stories and plays. Professor Epstein won the ACM SIGCAS Outstanding Service Award in 2003 for his contributions to computer ethics and the social implications of computing. "The Wheel" article and many of his stories and plays are available at www.cs.wcupa.edu/~epstein. He can be reached at RGEpstein@aol.com.