Interviewee: Mario Barbacci (MB)
Interviewers: Michael Davis (MD), Anthony Spencer (AS)
Location: SEI, Pittsburgh, PA
Date: 6-11-02

SOFTWARE DEVELOPERS WRITE A CODE OF ETHICS

QUESTIONNAIRE

1. *What is your educational background?*

MB: I have an electrical engineer's degree, from the University of Engineering in Lima, Peru, and a Ph.D. in computer science from Carnegie Melon in 1973. And, after I completed the degree I was hired to the research faculty at CMU, and I have been here since—as an employee—since January 1, 1974.

MD: Very stable.

MB: Oh yeah…I have a home here.

2. *What sort of organization do you work for? What do you do there?*

MB: The Software Engineering Institute was created by the department of defense as a result of a recommendation from a study group in 1983. Basically, the situation was that, up to that point, hardware was the dominant thing. So they had all these weapons and computers, and hardware was the thing they focused on. But, little by little, they were getting into the software business. There was a lot more software driving all of the systems; it wasn't just the hardware. And they were beginning to have problems and delays and cost overruns. So, there was a study group—they organize this periodically to discuss some particular topic. In

1983 there was a study group that looked at the problems the department of defense was having with software, and recommended that the department of defense create was is called a federally funded research and development center (FFRDC). FFRDC's are a special corporation. They are created by act of Congress, and they are dedicated to work with one of the U.S departments or units. So, for example, Cal Tech's jet propulsion laboratory is an FFRDC that works for NASA. Lincoln Laboratories at MIT is a FFRDC that works for the department of defense. They are very old—they started right after World War II. So there are probably over 50 or 60 of those. They are all created with a specific charter from Congress, and there is some department—department of defense—or some unit like the Navy—that oversees the activities in the unit. So, the recommendation was that the department of defense needed to create a software engineering institute dedicated to looking at software problems that the department of defense was having. The department of defense would oversee this. They also wanted a competition, so they asked the department of defense to draft a request for proposals so that a number of universities—I don't remember the details exactly…but I think the recommendation was it has to be run by a university. There might be industry partnerships, but they wanted somebody independent because many of the companies that might be bidding to be the software engineering institute might also be contractors, so they would have a conflict of interest. So there were a number—around 20 or so—

universities and groups of universities that bid for that. We were chosen as the first bid, and then there was some negotiation…because usually what happens is you rank them…you negotiate with the first one, and if you find an agreement, then they get the contract or you go on to the next one. So, this was happening in the fall of '84. We started on January 1, 1985.

MD: Okay. Originally in '85 it was given the name software engineering?

MB: Yes, from the beginning. In fact, the recommendation said the department of defense has to create a software engineer institute.

MD: So, what do they mean by software engineering?

MB: The term is interesting. The word engineering is in some states reserved for people with specific educational degrees, but it's not universal. The first time that the term software engineering, as opposed to software coding or developing, [was used] seems to be the NATO conferences in the late '60s. I think it was 1968. There were two of those conferences—1968 and 1969. But the first time that the term was ever used was at one of these NATO conferences in the late 60s. During the '70s you didn't hear a lot of references to that term, but it was in the community. And, when the recommendation came, there was a recognition that this was an emerging engineering discipline. And, they said if that's what you want to call it, you want the institute dedicated to that. It wasn't software science, software programming, or anything like that. It was an engineering discipline. The request for proposals was interesting because it had a number of candidate problems, and so the proposal that the bidders had to provide was how would they address this particular collection of problems. What was interesting was that the problems were not all technical bits and bites kind of things. One of the problems, for example, was about legal issues and licensing. In fact, it required that whoever won the contract had to work with lawyers. It had to do with the department of defense getting in trouble with licensing and rights to software. They would pay a contractor, and all of a sudden the contractor would deliver the software, and then the department of defense needed to do some updates and they discovered that they didn't have the rights to touch it. And so they had to go back to the original contractors. So, one of the—in the first year we had a number of lawyers, mostly academics from the University of Pittsburgh Law Department. CMU doesn't have one—

MD: That's across the street.

MB: Yes, right across the street. Exactly. The lawyers who looked at licensing issues we experienced in how intellectual property rights are managed. And, in particular in software, it's very easy for hardware devices, but how do you patent or copyright software? I mean, that was going back to the early…1985, the first year of the institute. And, the results were on the form of recommendations…technical reports to the department of defense.

MD: So that's still one of the things you do primarily, technical reports for the department of defense.

MB: Yes, but they are public. And, in fact, we work a lot more with industry than we do with the department of defense proper. Because the department of defense has changed. They have now very little internal development activities. Most of the software development testing and maintenance is done by private contractors. And so we tend to work with industry, and we provide reports which tend to be public. Very few things are kept under wraps. Only when they involve proprietary issues. For example, we might work on some contract on some acquisition issues or we learn about what the contractor might be providing and we might give some advice to the acquisition agency. Those reports would be kept confidential.

MD: Have you in particular used the term “software architecture”?

MB: Yes.

MD: So somebody would want you to essentially be writing the specifications for development of the piece of software.

MB: Right. Typically what happens is that we are approached by some unit of the Department of Defense. Some part of the Army, some Army project, who will ask us to conduct an evaluation of something that is being proposed by a contractor or they will ask us for some assistance in developing the request for proposal. Because it becomes a very touchy thing once you have a request for proposal out, you cannot touch it. It’s a very contentious thing, so they want to make sure that they ask the right questions there, and ask for the right documents in the proposal. So we get involved in lots of pre-acquisition projects as well as projects that are already awarded. There is a contractor in place, they are maybe two years on the project and the department may ask us to go and do an evaluation. We do this with a number of units: Army, Navy, Air Force, not just Department of Defense; we have a large project working with the Coast Guard, which is the department of Transportation. IRS at some point. Etc.

MD: And when you went out to do an evaluation do you just evaluate the process, or do you test the software?

MB: No. That is the difference. There is a group at SEI that does process measurements and what they analyze is the ability of the organization to carry out software projects. And they do have interviews, questionnaires, to assess that. What we do is not look at the organization but look at the projects and the product. And we ask for a presentation of the business writers. What is driving that organization to develop that software? Sometimes the business representation is provided by somebody from the user—say, the Coast Guard needs a system that doesn't fail during a search-and-rescue operation. Sometimes it is the contractor or a company that is trying to sell a product and will say there is a market for this kind of product, this is the business driver. Then we ask for the developer, the head of the team, usually the architect of the software. We ask them, okay, explain to us what properties your system will have, what are you producing, what kind of components, and so on.

And then we ask the stake holders in the system, we invite the large group of people—some of them are users, some might be maintainers, some might be customers of this system—and ask them: what do you think…would be important for you? And we ask them to propose a scenarios describing what this system will do, transmit messages in less than two seconds from point A to point B or whatever. Whatever matters to them. Then we prioritize those scenarios and ask the architect present while this is all happening, 'Okay, look at the scenario number one, explain how your system will do that.' And, sometimes the result of that is 'Gee, the system is unable to do it…maybe we need to change something in the software or we haven't thought of that particular component of the system.' So we don't know for example if those two components will use the same parameters and they were able to talk with each other. Sometimes issues might be postponed but what is very important is that we discover and comment on any misunderstanding between the stakeholders. It is always a social cultural problem. People don't talk to each other and having a meeting where we ask all of the variety of stakeholders to propose the scenarios is a very important way for them to communicate. We have seen lots and lots of these meetings where the architect is surprised that the users of the system had completely different intentions of the system. So the architects say, 'Oh we're going to do this thing that is just super fast, and the user says I don't care about the speed; I want it to be very easy to use.' You know like readable screens, and the architect might not have thought of that. Or the business driver is driven by his boss bragging that the system is going to give us 20% of the world market, and the architect is scratching his head and says, ‘Oops, that’s not what we’re doing.’ So, we tend to notice that most of the problems in the projects that get delayed or postponed, or go way over budget have to do with lack of communication between developers and stakeholders. The people that work on this project all have advanced degrees in computer science or software engineering and they are all techies and we’re discovering that the problem is lack of communication. Technology is not the problem.

MD: So, basically you’ve become a social engineer.

MB: It’s a social engineering thing. People don’t talk to each other. People don’t express their requirement or document what they are doing. It goes on and on and you see that continuously.

3. What experience, if any, have you had in software development?

MD: You’ve already answered that, but are you a software developer?

MB: No, I was always a researcher and all my time in the faculty so when I had to develop software it would have to be something that we were prototyping. I started my career in design automation so I was writing software to design computers. So, yeah, I developed software for a number of years but it was just for a specific internal project. It was nothing commercial.

MD: So the process was that you got an engineering degree?

MB: In Lima, right.

MD: You got…

MB: a Ph.D. here at Carnegie Melon in computer science.

MD: And your interest was in interfacing…

MB: Right. Initially my thesis was about design automation, and then over the years what tended to happen was I was interested in designing hardware but all I was really doing was writing software. No surprise you see a number of companies and organizations that initially were building hardware devices, you know, automobile companies. I know, because we do work with some of them. Traditionally everybody thinks well, automobile is about nuts and bolts and people, but you would be surprised how much software is in the modern cars.

MD: I hang around with mechanical engineers at IIT and a lot of them replaced parts of cars with software, various mechanical measuring devices. So I have a sense of what is happening. One thing that we're kind of interested in is what their origins are, how they end up in this field.
And are you an engineer? Without quotes, not a "software engineer", but
an engineer now?

4. Are you an engineer?

MB: Oh, that is an interesting question. I am in a boundary. A lot of the things I do, are engineering-like activities. We work with customers, we have a specific deadline so, I don't tend to work as a scientist exclusively. I’m not in some ivory tower, or working on far out projects. The things I tend to do are very specific. Lots of deadlines because we’re working with customers, so in that sense it is of an engineering activity.

5. *How did you hear about the IEEE/ACM Joint Task force on Software Engineering and Professional Practice (SEEPP)?*

MB: Oh, that’s interesting. I’ve been a member of both organizations for many, many years as well as something else called IFIP which is the International Federation of Information Processing. I tended to work as volunteer and officer in various organizations. So, I’ve been involved in professional activities for many years. It was early on in my career and that was something important to me. In the late 70’s I was part of a group that formed a new IFIP working group and become its first chairman. In the late 80’s I was approached by a friend to run as a candidate for the board of governors of the [IEEE] Computer Society. I was elected to the board of governors of the Computer Society and I had, beginning in the late 80’s, a number of positions. I was the president of the Computer Society in 1996. Vice president for technical activities prior to that as well. When the idea of having a committee to look into the creation of a software engineering profession, that was initially proposed at a meeting of the board of governors—I believe it must have been ’92 or early ’93—by a gentleman named Fletcher Buckley. You will see his name there. It shows up in some of the early…unfortunately he died maybe a couple of years later.

MD: He wrote papers on professionalization of software engineering?

MB: Right. Fletcher is the one, he was a member of the board of governors and he is the one who essentially started pushing the board of governors to create a committee to look into that. I think it was May of ’93 that the Computer Society board passed a resolution. You will confirm the date that…May 21, ’93 is when the board of governors passed that resolution that we’re going to form that committee. The ACM just a few weeks later passed a similar resolution saying that we need to look at that. We’ve always had lots of connections with them, joint members; so, it was not a surprise that we were on the same wavelength. So we decided that instead of doing two independent things, we will create a joint committee. So, from the beginning it was a joint committee of the two societies. I was appointed as the chairman as the first, initial committee. Later on I had to step down when I was elected President of the Computer Society because there were going to be too many things on my plate. But I ran that committee for a couple of years and that is when we started the joint task forces: one of them dedicated to the code of ethics; another one looking at the conducting of a survey of software engineering practices; another one looking at software engineering education. So, I started my involvement with this activity while I was the first co-chair, if you will. There was an ACM person [Stu Zweben]…we were both working together. We had an equal number of members present in these two societies.

6. What led you to participate in SEEPP's work?

MB: Mostly as an overseer of that. I wasn't really part of the committee. The steering committee was a small group, only 8 people. Three members from each society plus the chair and co-chair. Underneath that we had three task forces, and each one had a number of people. There was no requirement that they had to have equal numbers of people from the two societies. Don Gotterbarn was appointed as the chair of the committee looking into the code of ethics, and he picked his own volunteers. At one point I remember early on in the project he was in England on a sabbatical of some kind and so he was doing a lot of work abroad and he had some of the English faculty involved in the project. I was never part of that group, but I would get these periodic progress reports from Don.

MD: This question was not designed for you, so let me rephrase it. What led you to serve as the chair of the committee? What were you trying to do? What made it worth doing?

MB: It was in the Computer Society, and I won't speak for the ACM. In the Computer Society it was very clear that software engineering was an emerging field—that, for example, we have a number of publications, a number of conferences. We organize over a 130 some conferences a year and we have 19 publications, magazines as well as transactions. And it was very clear that the topics were evolving. The Computer Society started in 1946, IEEE goes back some 100 years. Initially it was mostly about hardware engineering. Over the years it was evolving and it was very clear in the early 90's that in fact most of our members really had something to do with software. That it was not people building metal components, they are not wiring things; they were coding things. So the necessity of defining that profession, the necessity of defining a curriculum that way you could certify that this person had the right training to become a software engineer was becoming very important. In the initial recommendation there is a description of that background. In fact, in most professions there is an accepted educational background. There is a curriculum that anybody in that profession understands. This is what the person in that field is to know. And this is how that person learns that. It turns out there isn't such a thing for software engineering. Not in an accepted way. There is a joint Computer Society and ACM curriculum recommended for software engineers, but it is mostly undergraduate curriculum updated every few years. It is good work but that is incorporated in the other task force on education. One of the things we noted in the very first report was that professions not only have this body of knowledge, this is what anybody in that field needs to know, and this is how to learn it. They also tend to have a code of ethics. That is an important aspect of being a profession. That there is a code of conduct that members adhere to. So, from the beginning there was the need to develop a code of ethics—Don might have more information on this—that was going to be more specific to software engineers. So we couldn't just adopt the ACM code that already existed or the IEEE code because they were very broad. We needed a code that focused on the kinds of problems that software engineers might face, the kind of situations they might encounter. Things like the conduct in the proper testing for something or reporting a particular problem. Initially, the program were used mostly by other engineers, so in the 1970's if the program had a bug, well, that usually meant somebody's experiment in the laboratory failed. Big deal. In the '90's, everybody and their brother are using the internet and sending email and many of them don't have a clue how the computer is working. So, all of a sudden you have to say, 'Okay, we don't want the person to have to know all about computers, therefore the person who is developing the software needs to do it right.' And that is where the need for a code of ethics focuses on the situation and on the needs of the environment where software engineers will work.

AS: How much of this was driven by concern for the public and software engineer's relationship with the public and their safety?

MB: Definitely. The need for the code of ethics…was one of the drivers. The reason you needed one is that software was becoming more and more visible and there were no clear rules for who could develop software, what they have to know and what were their responsibilities if something failed. There was nothing. And comparing that with other professions, lots of other professions, they all had some code of conduct. And so it was very clear from the get-go that we need one but we didn't know what that code of conduct will be. It was all Don's work and his committee who focused and decided that this was going to be important and the particulars in the code. So the steering committee didn't give any guidance or specific instructions. This all came directly from the working committee; the instructions to them were four lines of text that said, we need a code of conduct. They did the rest.

7. *Were you familiar with codes of ethics before you became involved in SEEPP? Explain.*

MB: Oh yes...as a reader. I have been a member of the IEEE and the Computer Society and the ACM for years, and one of the things that the two organizations do is once a year when you renew your membership you get a copy of the code of ethics. It's about a page or two and I've read them in the past…they can be very general. I'm not an expert in codes of ethics; I know that the American Bar Association has one but I don't know what it says. I'm sure the American Medical Association has one but I don't know what it says. So it's not my field of interest, so there was nothing I could provide as guidance. I was mostly a conduit. Don would communicate with me. He would send me the new releases as they were developed in the code of ethics, and I would arrange that they would be posted on the Computer Society web site. But I wasn't really involved in any of their meetings.

8. *In what ways did you participate in SEEPP's work, especially in the process of preparing the code? (The more details, the better.)*

MD: I think you just answered most of my question; basically, you received the documents and posted them and kind of watched…

MB: Right, and watched. And arranged briefings so once in a while Don would have to brief me and the board of governors. So I will schedule those briefings.

MD: So you were present at the briefing?

MB: Yes.

MD: Can you give us a picture of what those looked like?

MB: They were mostly status reports. There was…only for Don to give a tutorial to the members of the board of governors. He would give them an advanced copy of the code, so they all had it and then he would describe the status…'we are working on the next portion and perhaps we're going to have'…he also tended to organize workshops or special sessions in various conferences in the field as a way of disseminating, you know, telling people this is coming, we want your feedback and so on. So, that would be one of the things that he would report to the board of governors.

MD: These meetings were something like the meetings that you run regarding software?

MB: Yes. Well, there were meetings that were run as the working committee and then meetings of the steering committee, and then the next meeting of the board of governors of the Computer Society would give a short briefing. Don and the other people who are running the other task forces would give short talks. Those briefings tended to be 15 or 20 minutes, not a big deal. Certainly, in the Computer Society, it was a given that there was a need for the code of ethics and Don had included lots of people from around the world, that was one of the important things that he did, we didn't want it to be a US or North American-based code because that would create problems. The IEEE, and particularly the Computer Society, sees itself as an international society. In fact, almost 40% of the members are non-US. The IEEE is probably a lower percent, but it is growing. In all of the societies, the numbers of members from abroad and certainly the Computer Society is a big deal. We have lots of conferences organized around the world in many countries and for some of the larger conferences there is a requirement that they rotate around the world. So, international issues were a big deal; therefore it was very good that Don had people from many different countries involved in the task force.

AS: What would you say are the advantages and disadvantages of the code? Just in general?

MB: This is interesting. In some professions in the past, it very often would be the case that some country would be such a dominant power that in fact that country would dictate the rules. What we notice in software is that it's universal. Everybody uses the same language…they communicate in English typically, and they use the same programming languages. So things like Java are very popular. They communicate and send upgrades and do testing very much the same way. Let me give you an aside, one of the reasons that I noticed this shared culture. One of my jobs in the Computer Society today is that I am a judge in what is called the International Design Competition. Probably you have heard of the ACM programming competition, they have it every year, mostly in the US, Canada. What the Computer Society has is a larger scope. So the students are all teams of students around the world, universities anywhere, undergraduates, are given kits that include both hardware and software components and they are given an assignment. They can pick and choose whatever pieces of hardware, software they can use. They are given a limited budget, something like a $200: you can buy $200 extra software or hardware so that rich countries don't get an advantage. And you have to build this system. This year, for example, we had 74 teams from around the world submitting written reports. We did this at the end of May. From these we pulled it down to 10 based on the written reports. So, at the end of the one-semester project, they send a report, 30 pages limit. We read them all; we'll look at the requirements and so on. What was surprising is that you read the documents and unless you read the cover and see the name of the university or the name of the country, or the names of the participants, very often you couldn't tell what country these people were from. Because even American students would write in broken English! (Laughs) So you couldn't tell. They mentioned the same technologies, the same testing procedures and they mentioned the fact that they were using the same IEEE standard. You couldn't tell. So, software engineering has become defacto international. No countries dominate in the sense of having the intellectual leadership. There is more industry in this country. But we've had, for example… a Polish university has been in all three competitions so far, and they are very good. The Canadian's are very good; McMaster [University] and Waterloo are always there. Couldn't tell this year…we had some Taiwanese university there. You name it…around the world.

AS: So that's an advantage?

MB: It's an advantage because every country will be able to develop and the industry can grown anywhere. Before we started this morning we were talking with Michael about the CMM five stages [quality levels in software development process] that is known all around the world. And companies, such as Motorola have a laboratory in India that is ranked one of the highest in the world. It's a universal thing. They communicate in English. Use the same programming languages, use the same testing methodologies.

MD: And when you use software from Motorola, you don't know what parts of it were done in India, what parts were done in England and what parts were done in the US because…

MB: They get integrated. It doesn't require people traveling from one country to another with boxes of metal. You transmit it overnight by the internet. So, for projects, pieces of components of a larger thing might be developed in different countries, they get integrated overnight and maybe you run tests every day and so on. The speed of communication makes the difference, that is, you might be the Chicago office but getting some stuff from Texas and some stuff from India.

9. *By what means did you participate? For example, did you participate by email, or by phone, or through face-to-face meetings, or by letter, or by informal conversation, or the like?*

MB: Email correspondence with Don. That was the usual way, so I would get the progress reports. And Don sometimes…he would have questions. It was mostly email, the occasional meeting, or at a workshop, but email was by and large the way we communicated. And I suspect that was the way he communicated with his team because they were around the world. So, not surprising—it was already the mid-late '90s—, so everyone had access to the Internet and email.

10. *Did any of these means of participation seem to work better than the others? Any seem to work worse? Which would you recommend as best? Why?*

MB: Oh, that's interesting. I wasn't present at the meetings developing the code of ethics, so I don't know how contentious they were. Don would be the better person to tell you about it. The steering committee level meetings were very business like. There were no big debates because all we were doing was awaiting the reports from the various task forces. In only one of the task forces I had a more active role. The task force for developing the survey on the body of knowledge. I was involved because that effort was falling behind schedule. It was revived because we got some DOD sponsorship. Someone in the DOD was very interested in having a body of knowledge so that they could educate their own people and the Department of Defense gave money to the SEI so the SEI could pay for the development of the survey so that got me involved in that task force. I suspect there might be some interesting discussions in the committees that were developing the code of ethics, but not at the steering committee level. But I wasn't getting that. There were other things and you might hear them from Mary Shaw (CS Department, CMU) maybe the same or different. The two societies tended to have different positions on some things. Not so much on the code of ethics…that was pretty much approved by the two boards…it was a no-brainer. The development of the body of knowledge was a bit more contentious because that was getting closer and closer to the idea of having a profession and perhaps a certification for software engineers. And, in the Computer Society it was always a done deal. The board of governors was always unanimous in the sense that, yeah, there is a profession emerging and we're going to have to have some kind of certification or accreditation mechanism for software engineers. The ACM was split. This goes back a number of years. In computer science, they are very afraid that the field of software engineering that started within the computer science community will be stolen by the engineering schools. And so they were always very afraid. They wanted something called software engineering because that is a big chunk of computer science and they don't want to give it up. In different schools of engineering, computer science is part of engineering, in other schools computer engineering is part of computer science…there was a fear that the school of computer science would be losing its crown jewel. In my own school, this was one of the concerns that people had. Mary Shaw was very vocal in that regard. She was opposed to the idea of having a separate profession. To her, software engineering is a subset of computer science.

MD: I have been reading her papers and I thought she was big on it growing into an engineering profession. But she actually wants it to be an engineering profession inside computer science? That's interesting. I had no idea.

MB: Because such a profession is not subject to typical engineering professions rules…they have accreditation boards and so on. In general, the ACM was always split. My first co-chair was Stu Zweben. He was later, about the same time I was, the president of the ACM, he's at Ohio State. He's a computer scientist and then after he became the president of ACM, they Dennis Fraily as co-chair. He was next to Stu and he worked for Texas Instruments. He was very much in favor of having certification; in fact, if you recall, a couple of years ago the legislature in the state of Texas passed that amendment and he was the consultant to the group. He works near Dallas-Fort Worth. So, that is coming. In the Computer Society, it was never an issue whether there was going to be a need for this kind of profession. It would be a profession and it would be like any other professions. In the ACM, they tended to have this split between people who were the academics in particular in computer science departments; this was a very scary thing; they would be losing their crown jewel and all of a sudden computer science would become like mathematics or physics. Unless you are in the big universities, physics is a very small department. That's a fear. But it wasn't universal; otherwise the two societies would have never been working together. So the ACM over the years has had this ambivalent relation on the certification side. Not on the code side, that was very smooth.

11. *Any events that particularly stick in your mind relevant to the process? (The more details, the better.)*

MB: Not on the code of ethics. That seemed to go very smoothly. I think Don was very good at socializing the process. Talking to people, conducting surveys, broadcasting what that group was doing. There were never any firestorms that I noticed, or people sending me messages, 'Oh my God, terrible, terrible, terrible.' He was very good at socializing this. That is his personality, so I'm not surprised that it worked so well. The other one, developing the body of knowledge, that was quite a bit more work. We were way behind…the survey 2 or 3 years behind schedule and nothing came out of it. We did the survey, published the report and nothing happened. Then we restarted the project again with a different approach. We are working with the University of Montreal, and we have a number of people there who are leading the effort. They have developed what they call the Software Engineering Body of Knowledge and they have done two or three interim editions. They have identified a dozen or so main topics, they asked for volunteers around the world who will take the task of refining, editing, and getting comments and impressions. It is the trial run edition, available on the net. It's a book that is about 60 or 70 pages that has all of the 12 topics and themes, this is what a person should know about this topic. It is being distributed for comments. In about a year or so they will collect all the comments and make a second edition and I think that will be the final, or first body of knowledge. Once you have the body of knowledge then you can influence the educational community because, if this is what people need to know, then we need to provide courses for that. Not all of them will be undergraduate level, some of them will be graduate level and some of it might be optional advanced education. So there is something: after they finally come out with the body of knowledge, there is going to be some integration efforts and, given that the standard curriculum is done jointly by the ACM and the Computer Society, there might be some future friction there.

MD: I noticed that the last time the accreditation people went through, ACM's accreditation system is now being operated by ABET. Is there someway to coordinate it instead of having it separate?

MB: Yes. For a number of years the Computer Society and ACM have run a computer science accreditation organization called CSAB, Computer Science Accreditation Board. Very often ABET and CSAB visits would be coordinated to reduce the burden on engineering schools that also had a computer science/engineering department. After many years of negotiation, CSAB was incorporated into ABET. So ABET now has grown; that's the **A**ccreditation **B**oard for **T**echnology and **E**ngineering. There is now a new commission under it which is the accreditation board for computer science. There was always a fear that the old CSAB would lose all power, that the engineers would just swamp them [the CS people]. But it has worked just fine. In fact, schools have to be re-accredited every few years and I am one of the creditors for that new commission. It is interesting because I have seen it right in the building process. Up to 3 years ago, it was all run by CSAB; by this year it will all be under ABET. The rule won't change, the criteria, the credit hours—it is all being integrated and it's going smoothly. So I see it as a working "B". For example, with the visit to the University of Texas last year, it was a joint visit. Because back then it was still in the integration. This year sometime in the Fall I will be doing another one of those visits and there will be only one committee involved. After that, they will have an integrated accreditation tem for computer science, electrical engineering, and all of the other engineering disciplines.

AS: Looking back on the time before the code was implemented, how would you say things have changed in terms of production and development?

MB: I haven't seen much difference. The code probably needs some punch behind it. It will have to be something that causes people to pay attention…. A lot of changes in the engineering discipline have been because of the lawyers. So that is my favorite anecdote…that in the 1850's there were lots of steam ships crossing the Atlantic. Very often they would blow-up. Boilers were too hot, fragile. You would lose the cargo and people. Lloyds of London was providing insurance they had to pay on when a ship blew up. They didn't lke that. So they forced the Maritime industry to develop some standard processes for building steam boilers and the steam ships. It was the lawyers who told the engineers get your act together. Something like that might have to happen in software engineering.

MD: The boiler codes are a major achievement in mechanical engineering. They're still maintained and it's worth going to a library and looking at them. You have no idea the scale of…I guess 40 feet long, details and specifications.

MB: Quality of the material, the type of coal.

MD: The exact placement of the thermometer.

MB: It was because Lloyd's just got on their case. They said they are

not going to insure the boat unless they followed the standards. Companies that didn't have the insurance couldn't possibly survive in business. So, what might have to happen is there might have to have been attempts, people have tried to sue each other, there have been some cases where software surfaces in the case. What happens so far is that lawyers don't seem to be very comfortable talking about software. So, they have a case where a judge passes a decision where this company did something terrible because they didn't follow the standards for software development. Or their developers lied about something, or didn't behave appropriately and didn't conduct tests, for example, and the judge might say they didn't follow this code of conduct. They didn't tell their supervisor they had a problem in this or that area. And all of a sudden people will pay attention. That hasn't happened yet. So right now, the way it seems is that the code of ethics is an academic topic. Academics know about it and talk about it, there might be another version coming in a year or two. By and large, industry is not aware of what we are paying attention to.

AS: So in other words, it hasn't affected the business?

MB: All it would take is, as I said, a case in which a judge makes a decision saying, 'The people who are guilty or are responsible for the damage because they didn't follow the recommendations in the code of ethics.' And this will be cited in the case because it's an official document for these two organized societies, so somebody could actually cite it. And, if a judge says these people violated the code of ethics and therefore they go to jail or pay a fine—that will get people's attention.

AS: Well, if that's not the case, the code is perceived as having no bite, no teeth, how do the people that are supposed to adhere to it see the code?

MB: Well, having no teeth in the sense that there is no mechanism to enforce it. The sanctions will have to come from a legal case. As an aside, England has a similar society to the IEEE. The Institute of Electrical Engineers, the IEE. They have a different role than the IEEE. In England, IEEE membership automatically gives you a license to practice. So, becoming a member is a bit harder than becoming a member in the IEEE. You have to have certain degree of education, certain courses, and once you are a member, you are a chartered engineer. And, you can practice the profession. In this country, the IEEE or the ACM don't have that. You don't have to be a member, whereas in England you have to be a member if you want to work as an engineer or developer. So, if the IEE in England, for example, has a code of ethics…then people have to follow that code of ethics. We haven't got to that stage here.

12. *Do you have any documents, paper or electronic, relevant to your participation in the process? May we have a copy?*

Mario provided a stack of documents (steering-committee level).

13. *Has your thinking about codes of ethics changed as a result of your participation in SEEPP's work? How?*

MB: Not particularly, no. I always thought it was a good idea that people know…must know what they're…and by and large the code is a lot of common sense. So I don't think it's like a great surprise for a developer. They might learn a couple of things…steps to follow. In particular, I suspect, even if it is adopted and everybody has to follow it…it will only affect people who are involved in safety critical software. You know, if you are working in an aircraft factory plant…you know, you're building software that goes in a 777. Those will be the kinds of things. If you're building software that goes into a video game…probably not a big deal. So, it will be separated by the industry in which they are working.

AS: How do you think clients see the code?

MB: I don't even think clients know there exists such a thing.

AS: So, if their expectation is to something that doesn't follow the code or goes against the code—

MB: You mean somebody who buys the software?

AS: A client that contracts a company to develop some software.

MB: What I suspect is that the people who are developing the software will develop software that meets their requirements. They probably don't care how they do it. You know, unless they stole somebody else's patent or something like that. They pretty much…they couldn't care whether they followed exactly all of these steps, that they did the appropriate level of testing, or the right communication between the right development teams. They couldn't care less.

14. What, in your opinion, is important about having a code of ethics?

MB: It's mostly educational…for the developers…knowing what their limits are. And, also, enabling them. Having a code of ethics that is widely acceptable will give the ammunition for somebody who is uncomfortable with something happening in a project…being able to point to the code of ethics and challenge the supervisor. Or going to the president of the company and saying, 'Oh my God, we're violating a particular rule.' So, it will enable people…it will give them that freedom to challenge something.

15. *Is there anything about your <u>participation</u> you are especially pleased with or unhappy about?*

MB: Very pleased with the code of ethics. It was always on schedule, and Don did a very good job. It was clear that this was not a couple of people working on a corner and doing their own thing. I mean, there was a lot of socialization into the process. And so that's why it worked. Even though the code is mostly an academic topic, for the time being, it was never the case that there were challengesfrom some other group around the world. It turns out that at the time Don started, there were other code of ethics projects around the world. I think some of them were from Germany.

MD: Really? I hadn't heard about those.

MB: I think I have it here. Let me see.

MD: That's interesting.

MB: I have a memo from one of these groups, and I forward that to Don.

MD: Was the person in Switzerland?

MB: It might have been…the name sounded like a German name. But it could be Swiss. When you talk to Don, ask him if he remembers the— Yes, there was this group…I have the emails…see the names? And this email was sent May 12, 1994. Two emails in the same day.

MD: No, none of these names are familiar.

MB: Okay, so you keep this, and maybe you can use this in your discussion with Don, because they seem to be…they were doing something similar.

MD: A task force for professional ethics.

MB: So, one thing to talk to Don about is what happened. This is a project that started around the same time—no, in fact, preceded Don's project—what happened to that particular activity.

16. *Is there anything about the* *<u>final code</u>* *that you are especially pleased with or unhappy about?*

MB: No, nothing,…I like the format. This might not be an interesting thing. I was looking at the use of the code, and the fact that they had essentially a one-pager version and then lots of details behind it, as opposed to having another 20-page document that nobody will read, because your eyes glaze over. I thought that was a good way to present the code…it was a one pager. And some of them might have been driven by both ACM and the IEEE codes. They tend to be that focused—one or two pages. Something you can carry in your pocket, like a book.

17. *Is there anyone whose participation in the process seems to you especially important? Explain.*

MB: Well, my main contact was Don. He definitely was very clever in the way he organized the collectivity. The socialization, the communication, allowing anybody to provide input. So, definitely that helped. It could have been a disaster. I mean, you could've had somebody with a big ego that says 'I know what I'm going to do' and draft his own code and then say to the rest of the world 'Do you like it?' We see a lot of projects like that, and Don was completely different…he was very social. No, it doesn't mean that the code is going to be acceptable practice in the future. So, there is a code…if it's going to be updated periodically, whether it becomes a useful thing, something that is kept in mind by software developers or is just another academic exercise, remains to be seen.

18. *Anyone who you think we should be sure to talk to? Explain.* (SEE QUESTION 17).

19. If you had been in charge of the process, what, if anything, would you have done differently?

MB: I don't know if I could answer that, because I was not very involved in the process. So I don't know if something could have been done differently. The results were very good, but I don't know if there were specific problems that Don had to deal with. He kept those from me. He would just send me the progress reports.

20. Is there anything we should have asked but didn't? Anything you want to add to what you have already said?

MB: No. Not that on particular. Remember that I had those three task forces, and this was the one that nailed it down and was ahead of schedule. There was never any friction…I do have concerns about the way we did the body of knowledge…how we defined the body of knowledge. A couple of bad starts, and we are now doing a different version. So, I still have concerns about the definition of the body of knowledge, because from there, that is where you're going to get this education. So, however people get educated to become software developers, at least there is a code of ethics in place. But, I'm willing to wait 20 years—if I'm still alive—to see whatever happened to it.

AS: What about the people that don't posses the appropriate qualifications? What should they follow or abide by? Is there a separate code for them to follow? Or is there talk of developing a code for them?

MB: That's interesting. So you're talking about different ranks of software developers?

AS: Yes, those with varying degrees of skill, knowledge, or ability.

MB: Yeah, in the body of knowledge that was one of the topic areas. What is the body of knowledge, and there are different degrees. That is, do you want everybody to have a Ph.D. in computer science? Well, no, that's impractical. Do you want everybody to just simply hack pascal code? Well, no, that's not very useful. So there is some back and forth. I don't remember if the code of ethics actually addresses the level of education. I think it says simply 'appropriate or adequate'…it doesn't point to specific rank of education. But, that's a good question. You caught me there by surprise. I'll have to go back to the code and read if it says anything about whether you have to have a particular level of education or degree. I don't think it will say that.

AS: So what about the people that don't fall within that camp or category?

MB: That's a good question. I suspect that if there is an engineer in charge of that, a group of people who are not bound by the code,…then the engineer is the one responsible for applying the code of ethics. So, yeah, people who are mainly doing part-time jobs don't have the education…and they might not be responsible for following the code of ethics. But then, somebody in the organization has to be in charge of these people. That is true, in fact, in some fields. I think in civil engineering, I believe, all you need in one company is that one engineer has that certification so that he can sign documents. But, that person becomes responsible for testifying. The person who signs it is responsible. There has to be somebody who is responsible for the project at hand. And the assumption is that the person is following the code, and is responsible for enforcing it.