N89 - 16313

534-61
167055
15P.
[illegible] 532

Towards A Software Profession

by Edward V. Berard, EVB Software Engineering, Inc.

"Between the amateur and the professional ... there is a difference not only in degree but in kind. The skillful man is, within the function of his skill, a different integration, a different nervous and muscular and psychological organization ... A tennis player or a watchmaker or an airplane pilot is an automatism but he is also criticism and wisdom."

Bernard De Voto
from "Across the Wide Missouri"
[1947]

"Liberty trains for liberty. Responsibility is the first step in responsibility."

William Edward Burghardt Du Bois
from "The Legacy of John Brown"
[1909]

"The absurd man is he who never changes."

Auguste Marseille Barthelelmy
from "Ma Justification"
[1832]

"A professional makes it look easy."

Source Unknown

"Old age and treachery will overcome youth and skill."

Julian Levi
Motto for the "65 Club"

"Computer programming," as we know it today, is a little more than 35 years old. You might even say that, as an occupation, it is in its "late adolescence." Programmers. themselves, have been known to exibit all the symptoms of adolescents, e.g., arriving at work at odd hours, dressing in a unconventional manner, spouting technical gibberish that is seldom understood by anyone other than another programmer, referring to themselves as "gurus" or "wizards," and an extreme loathing to accept anything that even vaguely resembles responsibility. These items may be collectively referred to as the "Real Programmers Don't Eat Quiche" syndrome.

To be fair, an increasing number of programmers have attempted to change their image. They have made it plain that they wish not only to be taken seriously, but they also wish to be regarded as "professionals." Even the term "programmer" has become passe´. Many programmers, and their companies, now refer to programmers as "software engineers." (Note that this change in nomenclature is seldom accompanied by a corresponding change in job description.)

Put simply, changing the image of "programming" and programmers is a "tall order." Both the software and the people who deal with it suffer from a severe case of "Rodney Dangerfield Syndrome," i.e., they get little if any respect. Hardware professionals often look at software as something that one "slathers on the hardware" to get the real product (i.e., the hardware) out the door. Even programmers have few qualms about stealing software. People, in general, have a hard time recognizing software as a product.

The attitude that "anyone can be a programmer" is still very prevalent in our culture. The only credentials one seems to need to call oneself a programmer are a general familiarity with the syntax of a programming language, a rudimentary knowledge of a text editor, and enough exposure to an operating system to invoke a compiler. Most programmers are totally lacking in skills such as software design, software testing, software maintenance, software quality assurance, error analysis, metrics, and configuration management.

Our work seems to be cut out for us. If we wish software professionals to be considered professionals in every sense of the word, two of the major obstacles we will have to overcome will be: the inability to think of software as a product, and the idea that little or no skill is required to create and handle software throughout its life-cycle.

Professions and Professionals

If we are going to address the issue of *professionalization*, we must first define what it is we mean by a profession and by a professional. A logical place to start is the dictionary. The 1979 version of *Webster's New Collegiate Dictionary* provides two common definitions for a **profession**:

1. "a calling requiring specialized knowledge and often long and intensive academic preparation," and

2. "a principle calling, vocation, or employment."

Unfortunately, the second definition more accurately describes the "software profession" as it exists today. There are, however, a small, but growing, number of organizations where the first definition is more appropriate. These organizations have found that an engineering approach to the software life-cycle is not only less chaotic, but cost-effective as well.

Our dictionary also provides two definitions for a **professional**:

1. "one that engages in a pursuit or activity professionally," i.e., one who conforms "to the technical or ethical standards of a profession," and

2. "engaged in by persons receiving financial return."

If you have any doubt that the second definition more accurately reflects the "software professional," of today, merely ask a software professional to list (or give a specific reference to) "the technical or ethical standards" of the profession. The Computer Society of the Institute for Electrical and Electronics Engineers (IEEE-CS), has made a good start at defining some of the technical standards for the software profession. The Institute for Certification of Computer Professionals (ICCP) requires that those who pass a written examination *and* "subscribe to the *ICCP Codes of Ethics, Conduct, and Good Practice*" may use the designation "CCP" after their names. These last two points illustrate that attempts already have been made to establish technical and ethical standards for software professionals.

We should also note that there will probably be a need for paraprofessionals in the software industry for some time. (Webster's defines a paraprofessional as "a trained aide who assists a professional person.") While we will acknowledge the probable need for paraprofessionals in the software industry, we will not discuss their required qualifications in this article.

Characteristics of a Profession and Professionals

If you were to interview a number of different professionals (doctors, lawyers, teachers, nurses, airline pilots, electrical engineers, and certified public accountants), you would find that their professions placed a number of requirements on anyone who wished to be considered a professional, including:

- *a minimal level of training for entrance to the profession* . Many professions require a minimum of a four-year college degree from an accredited institution. Even those that do not specifically require a college degree often require many hours of training which may take years to accomplish.

- *some form of formal certification.* The classic examples are the bar examination for lawyers, the CPA examination for accountants, and board certification for medical doctors. *In many professions, certification is not a one-time affair, with professionals having to re-ceritfy every one to three years.*

- *some form of continuing education or training* . Just because a professional has acquired a college degree does not mean that he or she is finished with formal education. Teachers, lawyers, nurses, and other professionals are often required to take a minimum number of courses per year to maintain their certification. (Even if college courses are not required, most professionals must keep current with their profession. Imagine an accountant who is unaware of the most recent changes in the tax laws, or a doctor who was not up-to-date on the latest findings on a particular antibiotic she was prescribing.)

- *some minimal level of proof of performance.* Proof of performance can take many forms. For example, it shows up as "publish or perish" for college and university professors, successful diagnosis and treatment for doctors, and the won/lost record for attorneys. *Professionals must demonstrate that they can practically apply the training required for their profession.*

- *conformance to professional standards* . Prior to admission to a profession, a candidate will probably be made aware of the standards (e.g., methodologies, metrics, and levels of quality) for the profession. The certification process will most likely test the candidate's knowledge of these standards. Upon being accepted into the profession, the professional will be expected to conform to the existing standards, and to keep abreast of any changes to these standards.

- *adherence to professional ethics.* Webster's defines ethics as "a set of moral principles or values." Professional ethics involve such items as the professional's obligations to his or her client, the social responsibility of the professional, the relationship of professionals to their employers, and acts which might discredit or degrade the profession.

- *the taking of responsibility and acceptance of liability.* Professionals take direct responsibility for their actions. The profession, as a whole, usually provides some form of established guidelines and acceptable limitations on responsibility to guide the professional. Professionals may also be required by some municipal, state, or federal laws to take on some amount of additional responsibility. Examples of professional responsibility can be found almost daily in any newspaper account of an event which caused harm to one or more individuals, or in which some law was broken. Classic examples include plane crashes, bank failures, and medical malpractice cases.

The profession itself typically provides a number of benefits to its members, including:

- *the establishment of a number of professional societies.* Professional societies provide a number of services for their members. (In fact, when people speak of a profession they are often referring to the professional societies for that specific profession.) They sponsor continuing education for their members, publish professional journals and other periodicals, provide a forum for the members to express opinions and influence the profession itself, and generally represent the interests of their members.

- *providing protection for its members.* One of the the most common defenses used in a malpractice suit is that the professional was "following generally accepted professional procedures and guidelines." Professions also provide guidance in such areas as rights of ownership, items which will directly affect current practices, and career advancement.

- *maintaining public respect for the profession as a whole.* The words "profession" and "professional" usually have positive connotations in the mind of the public. This is no accident. Professions (both the professional societies and the membership in general) continually strive to maintain, and improve, the image of the profession in the mind of the general public. *This translates into increased status and financial gain for the professionals themselves.*

Cook and Winkle (in their book, *Auditing Philosophy and Technique*) observe that: "Professions are characterized also by the performance of intellectual services, as contrasted with manual and artistic labor. In addition, professions recognize a duty of public service and adopt a code of ethics generally accepted as binding upon their members." Later, in the same book, they make an interesting observation regarding the American Institute of Certified Public Accountants (AICPA): "Frequently, state regulations are modeled after AICPA pronouncements. Many court decisions use the statements from the AICPA as criteria to evaluate public accountants and their work." *This is a powerful statement.* It illustrates a precedent for a profession to directly influence outside governmental regulation of the profession.

Professionalization

Professionalization may be loosely defined as the establishment, and adherence to, a professional model where one either did not previously exist, or was not firmly entrenched. In a professional model, the individuals who refer to themselves as professionals exhibit most, if not all of the characteristics of professionals mentioned in the previous section. Further, there is a "professional atmosphere" which is established jointly by both the individual members of the profession and the existing professional societies. This professional atmosphere exhibits the characteristics of the profession, also described in the last section.

A number of steps occur in the process of professionalization. These steps need not occur sequentially (i.e., some may occur concurrently and some may even occur "out of order"), and (ideally) they should make use of work which has already been done. The following list of steps is usually required for professionalization:

1. recognition of the need for professionalization. This usually occurs when the need for highly-skilled, uniformly-trained individuals is recognized, i.e., the work performed by those already in the field becomes increasingly critical in nature.

2. the formal definition of the profession. This will include establishment of standard terminology for the profession, creation of job titles and descriptions for profession members, and identifying relationships (e.g., profession to profession, professional to client, and the relationship of the profession to the general public).

3. the identification of key professional societies. These societies will hopefully have already achieved some degree of formal status (e.g., the respect of the professional community, the publication of useful periodicals, and the conducting of local and national meetings). These societies will prove invaluable in aiding the rest of the professionalization process.

4. the establishment of minimal entrance criteria. This must include such issues as: minimal formal education, minimal experience (apprenticeship), and a certification process.

5. the establishment and recognition of education and training programs. These can include existing college curricula and profession-approved continuing education programs.

6. the establishment of formal certification (and re-certification) procedures. The certification process must be based on the minimum amount of useful skills and knowledge required by a "typical" professional. The re-certification process should be directed towards the career paths available to the professionals. The certification and education processes will obviously directly affect each other.

7. the collecting and promulgation of professional standards. Professional standards encompass such items as: procedures, methodologies, metrics, acceptable performance levels, and tools.

8. the identification of relevant professional ethics. A code of ethics for the profession must be established, made known, and adhered to by the profession. Ideally, the profession's code of ethics will be incorporated into the certification process.

9. the establishment of minimal, quantifiable performance goals for the profession and the professionals. Professionals must be expected to meet (and hopefully exceed) some minimal set of performance goals to maintain their professional status. They must have some way of knowing how well they are doing in their chosen field. The profession must continually strive to improve itself as a whole (e.g., a decrease in the average error rate per member).

10. the identification of required continuing education (and a continuing education process). No professions are static, especially the technical professions. The useful life of much professional knowledge continues to shrink. For example, it has been said that the technical knowledge of the human race doubles every four years. Typically, professionals are required to take a minimal number of prescribed courses per year to maintain their professional status.

11. the identification of professional responsibilities and liabilities. Professionals must be aware of their responsibilities to their clients, their employers, other professionals, and to the profession in general. Further, they must be aware of the liabilities which come along with responsibility. Only people who are legally insane are not held accountable for their actions.

12. the establishment of mechanisms for filing of grievances, removal of individuals from the profession, and appealing both. The status of a profession is diminished by the inclusion of individuals who no longer meet standards established by the profession.

13. the establishment and maintenance of a positive public image. A profession with a positive public image can command better benefits for its members, including higher levels of compensation.

While all of the above obviously take time to occur, they can be accomplished in a relatively short time, say within three to four years. This time can be further shortened by a focused effort on the part of the professionals themselves.

Examples of Professionalization

Examples of professionalization abound. Outside of the software industry, we have as examples: the legal profession, the accounting profession, the medical profession, and the teaching profession. The professionals in these areas are lawyers, certified public accountants, doctors and nurses, and teachers respectively. Even a casual conversation with any individual associated with these professions would reveal that most, if not all, of the points covered in the previous section are relevant to their profession. The mechanisms and the nomenclature may vary from profession to profession, but the points themselves still remain relevant.

Professionalization occurred in these, and other, disciplines for a number of reasons. In some cases, it was the desire to disseminate knowledge, skills, and techniques in a uniform manner. For others, professionalization was forced (or threatened) by some form of government. Quite surprisingly, professionalization does not seem to occur at any consistent time in the existence of a discipline. For example, it took literally thousands of years before civil engineering became professionalized in the modern day sense, while electrical engineering became professionalized almost as soon as it was recognized as a separate discipline.

A major factor in the speed with which a discipline becomes professionalized seems to be the environment in which it functions. For example, when electrical engineering first came into existence, other scientific and engineering professions were already firmly in place. These established professions provided paradigms for the creation of the electrical engineering profession. When one views the handling of software (and related issues) throughout its life-cycle as an engineering problem, we can easily see that paradigms already exist for the professionalization of those who are directly responsible for software. Further, this professionalization process should be taking place now, i.e., definitely much before 1990.

Some of the steps necessary for professionalization are already in place. We have a number of professional societies. Computer science curricula at colleges and universities have been in place for more than twenty years. (Although some schools offer some "software engineering" courses at the undergraduate level, it appears that none are offering undergraduate degrees in software engineering, and only a few are offering advanced degrees in the topic.) The IEEE-CS is actively involved in defining standards for the engineering of software. The Institute for the Certification of Computer Professionals (ICCP) and the Certified System Professional Program (CSPP), among others, have established model programs for the certification of software professionals. The Ada Joint Program Office (AJPO) has established a working group in Ada and software engineering education. One of the issues that this group is looking at is the certification of Ada professionals. Finally, the Europeans are also exploring the idea of certification of computer professionals.

Software Engineering and Computer Science

For purposes of this article, I will restrict our attention on professionalization to three general categories of potential software professionals: computer scientists, software engineers, and software engineering management. It is not our intention at the moment to provide in-depth definitions of each of these professions. Instead, we will provide quick sketches of each, and leave the details to a later article.

In discussing computer scientists and software engineers, we will differentiate the two using the paradigm of more "conventional" engineers and scientists. A scientist is chiefly concerned with explaining current phenomena, predicting future phenomena, and generally improving the state of technical knowledge available to the human race. An engineer takes the information supplied by scientists, and others, and uses this information to produce cost-effective, paragmatic solutions to real-world problems. (These are obvious oversimplifications, but they will suffice for now.)

A computer scientist might be looked upon (most simply) as an applied mathematician. Some would rightly say that, with the disappearing differences between hardware and software, that some areas of computer science encompass a good deal of computer hardware technology. For our purposes, a computer scientist has a minimum of a four-year degree in an approved computer science curriculum (e.g., the ACM 1978 curriculum) from an accredited college or university. (Remember, the degree alone is not enough to make one a professional.)

A computer scientist might specialize in compiler design, queueing theory, operations research, or language-driven hardware architectures. While a computer scientist might focus on the details of some aspect of computer science (e.g., algorithm design), he or she might not have an immediate practical application for the technology they are uncovering. It is very likely, though, that this uncovered technology can be used in some existing, or soon to exist, practical application. (Just as a physicist studying the quantum mechanics of molecular collisions might produce results which have applications in gas lasers.)

Just as "conventional" scientists build on the work of other scientists, a computer scientist most often builds on the work of other computer scientists. *This means that an error introduced via carelessness or faulty analysis on the part of one computer scientist can have drastic consequences even outside of that computer scientist's immediate area of specialization.* Keep in mind that software permeates our very existence. (It has been said that the average American comes into contact with at least "two dozen computers" every day.) In addition, computer science technology is being used in increasingly critical application areas (e.g., pacemakers, cruise missile guidance systems).

Software engineering, like any engineering discipline, involves a mixture of technologies. The basic background of a software engineer requires: computer science, mathematics, engineering disciplines (e.g., design methodologies, metrics, error analysis), communication skills, imagination, problem solving skills, ingenutiy, and a respect for simple, pragmatic solutions to real-world problems. Software engineers, like their "more conventional" counterparts, has a minimum of a four-year degree from an accredited college or university. Unfortunately, the few software engineering degree programs that currently exist are almost exclusively graduate programs.

Software engineers may be charged with any number of tasks, e.g., the developing, testing, maintaining, measuring, or assuring the quality of a particular software system. They constantly find themselves integrating different technologies, making tradeoff decisions, and dealing with many different types of people (users, other engineers, managers). Most of the time they have a very tangible goal in sight. This goal must be reached within the (often unreasonable) limits on time, money, and other resources established at the begining of the project.

Like computer scientists, software engineers build on the work of other software engineers (*with the same serious implications*). Like computer science, software engineering is a rapidly growing, rapidly changing discipline. The software engineer is being asked to apply his or her skill to increasingly critical applications, e.g., life-support systems or the Strategic Defense Initiative ("Star Wars").

Even with the best-trained computer scientists and software engineers, the success of a project is not guaranteed. Poor management can ruin any project. The benefits of a truly professional technical staff cannot be realized without the support of professional management. While a successful professional software project manager need not be a technical wizzard, he or she must know who to hire, what to ask for, and what can reasonably be done with existing technology. We must be just as concerned with the professionalization of technical management as we are with the professionalization of the technical staff.

As you can see, I have described a huge task: professionalization of software personnel, i.e., computer scientists, software engineers, and technical managers. I want to now further narrow the scope of this discussion to the professionalization of software engineers. This is done primarily to keep this article from "becoming a short novel." However, much of what we have to say about software engineers will hold true for computer scientists and technical managers.

The U.S. Department of Defense (DoD) has advocated a software engieering approach to their Ada® effort. In fact, Ada is but one small piece of the DoD's Software Technology for Adaptable Reliable Systems (STARS) effort. We will use this a basis for our discussion for the professionalization of software engineering. Virtually everything we have to say will apply to the professionalization of all software engineers, regardless of whether they use Ada or any other language.

The Motivation for the Professionalization of Ada Software Engineers

There are several people who have a vested interest in the professionalization of Ada software engineers:

- the contracting office,
- the software engineer's employer,
- the software engineers themselves, and
- the general public.

One of the most difficult tasks for a contracting office is determining the *real* capabilities of a potential contractor. A university degree is somewhat meaningful, but often the contracting office is more interested in the actual "on-the-job experience." On-the-job experience is usually measured in the types of projects the personnel have previously been associated with, and the length of time the personnel have logged on each project. These are, unfortunately, very crude metrics.

If software engineers were professionalized, however, the contracting office's job would be somewhat easier. For example, if a potential contractor identified an individual as a software engineer, the contracting office might be able to make the following assumptions about that individual (depending on how the profession and its professionals have been defined) :

®Ada is a registered trademark of the U.S. Government (Ada Joint Program Office)

- *he or she has had a minimal amount of education at an accredited institution* . Further, this education covered a specific set of known topics which were directly relevant to their job.
- *he or she has known professional standards and guidelines to follow,*
- *he or she has gone through some known form of certification (and re-certification) process,*
- *he or she must abide by a known set of professional ethics,*
- *he or she has had to exhibit some minimal level of performance in order to remain in the profession,*
- *he or she will take direct responsibility (and liability) for their work,* and
- *he or she will be required to take a minimal amount of continuing education each year.*

In essence, the software engineer becomes more of a known quantity. (*It is important to realize that professionalization guarantees only minimal levels of quality.* While this might not seem like much, remember two things. At present there are no guarantees of any level of quality for any software "professional." Second, establishing a "floor for performance", tends to raise the "ceiling of performance" for the profession as a whole.)

The employer of the software engineer has a number of reasons for being extremely interested in the professionalization of software engineers, including:

- *all of the reasons listed previously for the contracting agency.* This makes hiring much easier.
- *while a professional might cost more than a non-professional they are usually much more cost-effective (i.e., productive) than non-professionals.* This does *not* mean that all non-professional software engineers are not very productive. It means, depending on the effectiveness of the professionalization process, that the odds are greater that a professional will be more productive because he or she will most likely have been exposed to productivity increasing techniques.
- *a professional is more likely to have a more mature, business-like (i.e., professional) attitude.*

Software engineers will, of course, be interested in professionalization. Some of the more important reasons, include:

- *the ability to know, in advance, the minimal criteria for entrance and advancement in the profession.* There will be a number of secondary benefits along this line. For example, it will be easier to coordinate college and university curricula with the demands of the job market. In addition, the requirements for advancement along a specific career path will be better defined.

- *the ability to determine how much an individual software engineer has improved over time.* At present, few software engineers know how they "stack up" against their fellow software engineers. They also have little idea about how to improve their status (worth) in their chosen field.
- *the chance to learn new things, which are relavent to their profession, on a continuing basis.* An active re-certification program will encourage (and obligate) software engineers to remain current in their field.
- *the protection and advice of the profession,*
- *known standards, guidelines, and practices which are established by the profession (i.e., not by some organization with little, or no, familiarity with current technology),*
- *the respect given to professionals, by the public, and by other professionals.*

The general public will be interested in the professionalization of software engineers for a number of reasons, including:

- *as taxpayers and consumers, the public is keenly interested in acquiring high quality software at the lowest possible price.* We should not have to belabor the point that software is consuming an ever-increasing chunk of every tax dollar, and of every new modem appliance.
- *professionalization reduces the chance of major (and minor) disasters which are the result of erroneous software.* If the general public had any idea how much of their daily lives, and their national security, depended on software, there would be an immediate large public outcry for professionalization.
- *the public, as a whole, is more comfortable dealing with professionals (e.g., airline pilots, doctors, lawyers).*

Who and What Needs To Be Certified

The item which will probably evoke the most controversy in the professionalization process is that of certification. Before we go any further we should define what we mean by certification. The certification process for the software engineers and technical managers themselves will probably be not unlike that currently used in other professions, i.e.:

- It will require that the candidates have a minimum level of formal education.
- Candidates may have to serve an apprenticeship (residency) for some pre-specified period of time.
- A written examination, possibly spanning several days, will definitely be a requirement.
- Personal and professional references may have to be supplied.
- The candidates will have to sign a document saying they will adhere to a professional code of ethics.

- Other documents that may have to be signed might address items such as professional conduct, acknowledgement of professional responsibility and liability.

- The candidates may have to demonstrate that they are covered by any appropriate insurance policies (e.g., malplractice insurance).

- If this is a re-certification process, the candidate will have to demonstrate that he or she has taken appropriate continuing education courses within the time limits specified by the profession.

If, for example, we focus on the Ada community, we find that certification can be applied to a number of items, including:

- Ada and software engineering courses,
- Ada and software engineering curricula,
- the instructors for these courses and curricula,
- the graduates of these courses and curricula (managers as well as technicians) and,
- the software, standards, and procedures created by, or used by, members of the profession.

How Can Professionalization Be Accomplished

We have previously discussed a number of things that will be necessary for professionalization. To assure that the process itself is as effective as possible, we will have to consider the following:

- *There must be some form of quality assurance for the entire process.* Transcripts will have to be verified. The quality and appropriateness of any written tests will have to be monitored.

- *Someone will have to track and analyze the results of the professionalization process.* For example, certified professionals will have to be interviewed to identify weaknesses in the system.

- *Industry, academia, and the government must be constantly polled for constructive feedback.*

- *The process must be updated in a regular and timely manner.*

The Impact of Professionalization on the Ada Education Process

The first large impact of the professionalization process will probably be in the area of Ada education. Why? The thrust of Ada technology is not the Ada language itself, rather it is the overall improvement of the handling of software throughout its life-cycle. An examination of the STARS effort shows that a large part of that effort is focused on the improvement of human resources.

Although Ada educators have been aware that Ada had something to do with software engineering, most have given token attention to the topic, e.g., they mentioned the terms "abstraction" and "information hiding" frequently during their courses, but failed to address topics like software quality assurance, testing, design methodologies, and software engineering metrics. *Professionalization will undoubtedly require an increased emphasis on software engineering, mathematics, and computer science in Ada curricula.*

One of the major mistakes made by Ada educators is the assumption that software engineering "will be taught in a separate course immediately prior to (or following) the 'Ada course'." Professionalization will require that software engineering (along with ethics, standards, and mathematics) permeate the entire Ada curriculum. *This will have a definite impact on the selection of instructors, and students, for these courses.*

Instructors will have to exhibit some qualifications in addition to a knowledge of the syntax of the Ada language. Indeed, the qualifications of an "Ada technology instructor" will be have to be quite varied. Probably the least important part of the instructor's qualifications will be the knowledge of Ada syntax. Further, these instructors will have to go through some sort of certification process before they are allowed to teach.

The students in a professional-oriented Ada curriculum will find that they must meet some entrance criteria before being admitted. In addition, they will most likely be graded during the course of their training, and may fail, i.e., not be given credit, even though they attended the training.

The Re-Certification Issue

Re-certification does not mean giving the same test over again. Neither does it mean giving a "slight" variation on the same test to an individual who has previously taken, and passed, an earlier version of the test. Re-certification involves two broad areas: the recognition that software technology is extremely dynamic, and that a software professional may wish to advance along any one of a number of career paths.

It has been said that, in 1963 the technical knowledge of the human race was doubling every *ten* years. In 1983, someone observed that the technical knowledge of the human race seemed to be doubling every *four* years. There is little doubt that our knowledge of software technology is increasing at a faster pace than technology in general. (Ironically, most software practicioners seem to be "stuck" somewhere in the 1960s in terms of the way they deal with software.) Therefore, like other professionals, software professionals will have to demonstrate that they are aware of the significant current trends in their industry. Further, they will have to demonstrate proficiency in some of this new technology, i.e., that which is directly related to their immediate job.

This can be accomplished in a number of ways. Software professionals will, of course, be required to take and pass a number of continuing education courses on a yearly basis. There will also have to be some way that they can successfully demonstrate proof of performance at their jobs. (As yet we have few metrics for this, e.g., management and peer evaluation.) An actual re-certification test will be required on a regular basis (possibly yearly, or every eighteen months).

Like any other professional, software professionals may wish to advance along a given career path, or change paths. Here one of the advantages of professionalization becomes obvious. The requirements for advancement along a given path, or for changing career paths will be well-defined (at least much better defined than they currently are). In the event that a software professional wishes to change career paths, the re-certification process will require continuing education and a re-certification test as before. However, the demonstration of proof of performance may now encompass an apprentice period in the area of the new career path.

The Difficulties Involved In Professionalization

Even these who are favorably disposed towards professionalization will admit that a number of barriers to the process exist. It is important to realize, however, that most, if not all of these barriers can be successfully overcome. Here are a few of the more interesting problems we can expect to encounter:

- *Deciding on methods, procedures, and metrics* will be one of the first obstacles we will encounter. The important thing to remember here is that there is no guarantee that we will recognize the best methods, procedures, and metrics when we see them. This means we will have to pick some "good-looking" trial examples and be prepared to change them.

- *The cost and logistics of professionalization* will be staggering. Yet even these costs will pale in comparison to the costs of ignoring the need for professionalization, e.g., the cost of malpractice suits and insurance.

- *The development of meaningful tests* for certification will be complicated by the number of areas for which they will be needed, i.e., we will have to develop separate tests for software engineers, computer scientists, and technical managers.

- *Defining and determining success* will be one of the biggest problems. Fortunately, we will probably have to define success as part of the justification process for professionalization. This means that this problem will be considered early (and solved early).

- *Extreme initial resistance from those currently involved with software* , i.e., current programmers and managers, will be a major (if not *the* major) difficulty we can expect to encounter. However, the impact of this difficulty will be lessened by two facts: most programmers and managers want to do the best possible job, and programmers are very goal oriented. Don't forget the advantages we listed earlier.

- *The interaction of certified professionals with non-certified management* will prove to be an interesting problem. For example, what recourse does a certified professional have when he or she is instructed to do something which violates existing professional standards or ethics?

- *Maintaining a high level of quality throughout the entire professionalization process* is yet another item. This will be a classical "who will watch the watchers" problem. It can be solved by appointing an independent group of quality assurance people at the begining of the process, and giving them the authority needed to accomplish their mission.

Alternatives to Professionalization

One of the more obvious questions is: "What happens if we ignore the issue of professionalization? Will it just go away?" Unfortunately the answer is no. In addition to not realizing all of the benefits listed earlier in this article, there are two main problems we will have to deal with: the very real possibility of government regulation of the software industry, and an increased impetus for the automation of the software process.

One of the main reasons the AICPA was founded was the realization that the federal government was seriously considering the regulation of the accounting profession. By accomplishing formal professionalization (via the AICPA) accountants realized two goals. First, the accounting profession was able to provide its own regulation, instead of being regulated by those with little knowledge of their industry. Second, whenever municipal, state, or federal governments must develop laws which directly or indirectly affect the accounting profession, they consult the AICPA. *It is not uncommon to see AICPA rules and regulations quoted directly in laws governing the accounting industry.* It would not be unusual to assume that the software industry could accomplish the same goals via its own professionalization.

The forces that shape technology are seldom technological. They are more often political, economic, or sociological. If the software industry generally refuses to advance itself through professionalization, the public may react by placing an increasing emphasis on automating as much of the software process as possible, e.g., the increased use of fourth generation languages and off-the-shelf software. This can, and will, mean a direct loss of jobs in the software industry. (This will occur anyway. However, we do not necessarily wish to accelerate the process.)

Recommendations

What recommendations are to be made? The following will serve as a starting set of recommendations:

- *We must begin at once with positive results to be visible within two years* . Specifically, the "average programmer" and the "average manager" should be affected by the professionalization process before the end of 1987.
- *We must solicit input from many source* s. Included must be programmers, analysts, managers, educators, government, professional societies, and other, more established professions.
- *The process must be publicized and highly visible.*
- *A professionalization maintenance committee must be established.* The job of this committee will include tracking changes to the professionalization process, introducing these changes in an orderly manner, and acting as a "supreme court" for any professionalization matter disputes.
- *Encourage the establishment of software engineering curricula on an undergraduate level.* Further, encourage the concept of professionalism on all forms of software education.