

*A computing visionary and leader of the movement to define and elucidate the "great principles of computing," Peter J. Denning is a professor at the Naval Postgraduate School in Monterey, California. He is a former president of the ACM.*

UBIQUITY: How much success do you think you've had advocating that computing is a science?

DENNING: I find little argument with the claim that computing is engineering, but skepticism toward the claim that computing is science. In the past few years there has been a sea change on the science claim. The skeptics are coming around. Part of the reason is that scientists in other fields, particularly biology and quantum physics, have declared that information processes occur naturally in their fields.

UBIQUITY: Why do you feel the science claim is important? Some people might wonder what difference does it really make. Could it perhaps be a distinction without an important difference?

DENNING: Four reasons. (1) It's important for collaboration because it establishes credibility with the natural science fields with which we work closely. (2) It's important for innovation because someone who can see what principles govern a problem can look for possible solutions among the technologies that conform to those principles. (3) It's important for the vitality of our field because it helps us clarify the big questions that occupy us. Today's big questions overlap fields, such as biology's question, "What is the information process by which the organism translates DNA to new living cells? Can we influence or manipulate that process to heal disease?" (4) It's important for the advancement of science because natural information processes and natural computations are being discovered as part of the deep structures of many fields; we need a common language to discuss these phenomena. The Great Principles of Computing framework is such a language.

UBIQUITY: How did this develop as a major interest of yours?

DENNING: In many ways, it has been an unflinching interest for 40 years. When I began to call myself a computer scientist in 1967, I encountered many people who reacted with the question, "What is the science in computers?" It was not easy to answer that question, and

little in my academic training helped. That question has always nagged me. My own experience of computing is that many issues are deep and fundamental. I've always been optimistic that we would find a convincing answer, since computer scientists have learned so many things about computing that are completely non-obvious but have major implications for what we do in the world. In the past decade I've felt that computing is mature enough to tell its story and answer this question.

UBIQUITY: And "fundamental"? Fundamental in what way?

DENNING: Fundamental means that it affects all of computing, in any field. Let me give an example. Computer scientists have defined a class of problems abbreviated "NP" (for nondeterministic polynomial), whose members share the feature that finding solutions can take centuries on the fastest supercomputers, but verifying solutions takes only a fraction of a second. This class includes over 3000 common problems from science, engineering, and commerce -- problems such as finding optimal routes, schedules, subsets, and classifications of data. Billions of dollars are spent every year computing solutions to these problems. We know of no fast solution algorithm for any of them. But we do know that a fast algorithm for any one can be converted to a fast algorithm for any other. They are all equally hard (or equally easy). If anyone finds a fast solution for any one, everyone benefits. This discovery of computer science helps us understand why intractable computational problems are encountered in so many fields; and it helps us find heuristics that solve them reasonably well in reasonable time. Now that's fundamental.

UBIQUITY: I see.

DENNING: As fundamental as the NP-problem principle is, it is not alone sufficient to establish computing as a science. If this were the only deep principle we could bring forward, we wouldn't have much of a science. But we've brought forward numerous other principles such as the uncertainty principle for selection among near-simultaneous signals (first discovered in computer architecture), the bottleneck principle (networks), the locality principle (operating systems), the two-phase locking principle (databases), and the hierarchical aggregation principle (systems design). There are many other principles besides these.

UBIQUITY: Does that settle the issue of whether computing is a science?

DENNING: No, it does not. Having deep principles is not enough. The cynics say that any field that calls itself a science can't be a science. The serious critics invoke additional criteria including non-obvious implications of principles and falsifiability of hypotheses. I think we can answer these. The most difficult objection has been that computing can't be a science because science deals with natural phenomena, whereas computers are manmade. They say that, at best, computing is a science of the artificial, not a real science.

UBIQUITY: And your response to these criticisms has been what?

DENNING: In 2004 I sat down and carefully checked how computing does or does not satisfy all the accepted criteria of being a science. These criteria include an organized body of knowledge, a track record of non-obvious discoveries, an experimental method to test hypotheses, and an openness to any hypothesis being falsified. They also include secondary distinctions such as interplay between science and art, and between basic and applied. I saw that we could check off every one of the accepted criteria. (I shared my results in an IT Profession Column in the *Communications* of ACM [April 2005](#).) Even so, I still found resistance to the conclusion that computer science is a science. The single remaining objection was that, down at the bottom, science deals with natural things. I quipped that people are found in nature and people build computers. No, that doesn't cut it, came the reply, science deals with things that existed before people came along. Recently, as natural scientists have pointed out natural information processes, this criticism has been answered.

UBIQUITY: Do you think social science is a science?

DENNING: Social science has been moving in the direction of satisfying the accepted criteria for being a science. In my opinion, they have a longer distance yet to travel. They have an organized body of knowledge, non-obvious discoveries, and an experimental method. Like computer science they suffer from the criticism that social interactions are not processes of nature; social scientists stoutly defend the naturalness of social processes, even as biologists study social interactions in communities of ants, bees, and animals. I'm willing to accept that human interactions are natural processes. The place where social science still falls down is in the falsifiability of hypotheses. That is, they make some claims that cannot be falsified by experiments.

UBIQUITY: Quick example?

DENNING: Opinion surveys. While opinion surveys yield data, we can't use an opinion survey to conclusively demonstrate that people believe something. They could be misrepresenting their beliefs. Social scientists need to be careful in the kinds of claims that they offer as scientific, so that they can meet science's falsifiability criterion. You know, computer science had this same problem in its early years. Artificial intelligence (AI) researchers claimed that the mind is a computational process. When they discovered the rules of the process, a computer that followed the same rules would be conscious and intelligent. This claim, often called "strong AI", is not falsifiable. In the past two decades, mainstream AI has focused on falsifiable claims, such as "a neural net can be trained to read handwritten addresses from postal envelopes." In the older terminology, this was called "weak AI", but it has transformed AI into a strong scientific enterprise.

UBIQUITY: What about the science of the artificial claim?

DENNING: The claim is that, because computers are artifacts, all the things we study about computers are artificial. The purists say that a true science deals with natural, not artificial, entities. They say that computer science is about programming, chips, networks, and other technologies, all of which are human inventions. These technologies won't necessarily be around in a generation or two. How can their study amount to a science?

UBIQUITY: And your reaction to dismissals of that sort?

DENNING: As I've said, they bothered me. They did not square with my experience. I did not accept them. I knew from my own experience that computing technologies exploit many fundamental principles that have been in play for two generations and will still be play in two more. It struck me that if people in other fields think we are mainly programmers and technologists, we have been grossly unsuccessful at communicating the full breadth and depth of everything that we have done.

UBIQUITY: What is our method of explaining computing? How has it failed?

DENNING: We have three favorite story-telling approaches. In the first, we extol programming as our fundamental intellectual skill.

Programmers are at the center of many stories, be they heroes or rogues. The difficulty with this approach is that many people now equate computer science with programming. As they see programming jobs migrate to other countries, they think that computer science itself is being auctioned to the highest offshore bidder. A second favorite approach is the claim that the essence of the computing field is abstraction. After all this is what we do when programming and is the basis for several of the most popular contemporary languages. It seems to be what we do when we bring what is called "computational thinking" to a problem. The difficulty with this approach is that people in many other fields see themselves as making and working with abstractions. There is nothing unique about computing in this regard. The stronger we make this claim, the feebler it sounds. A third favorite is to tell the stories of computing technologies -- the silicon chip, the operating system, the Internet, or the expert system. The difficulty with this approach is that it ties computing to rapidly changing technologies. It sounds like we are defined by ephemeral inventions. Who can be sure that any of these technologies will be with us in a generation? And the field that generates them?

UBIQUITY: Is there objective evidence that this is how we see the field?

DENNING: Sure. Just take a look at any college catalog. The computer science departments all advertise the field in the same way: developing the skills of programming, working with abstractions, and covering a range of computing technologies. Take a look at the body of knowledge in the ACM/IEEE curriculum [recommendations](#): they portray the field as 14 main headings, mostly technologies, covering about 130 subheadings. The consensus view of our field emphasizes programming, abstraction, and technologies.

UBIQUITY: So you're saying that our failure to communicate comes from a habit of mind rather than a defective story?

DENNING: Exactly. I'm not saying that this way of expressing our body of knowledge is wrong. It communicates well when our primary audience is technology-minded people like ourselves. But today computing affects many people in all walks of life. Our primary audiences listen for principles deeper than technologies. We can't sell our field to others simply by hiring good journalists to tell our technology stories. We have to be willing to tell our stories in a different way. We have to find ways to discuss computing so that our

listeners can see their own struggles in the stories and then see how computing can help them.

UBIQUITY: Where did this line of thinking lead you?

DENNING: It seemed to me that if we are a science, we should be able to explain our fundamental principles. I began to ask what they are. I discovered that neither my colleagues nor I could say. I grew up with the definition that computer science is the study of phenomena surrounding computers. This definition puts the computer at the center. It holds that computation is what computers do. In 2001, when Biology Nobel Laureate David Baltimore said that cellular mechanisms are natural computational means to read DNA and construct new living cells, I saw that our definition, and the thinking behind it, is backwards. Computation is the principle, the computer is simply the tool.

UBIQUITY: But how can you have computation without a computer?

DENNING: Our older definition is like saying that biology is the study of phenomena surrounding the microscope, or astronomy is the study of phenomena surrounding the telescope. How odd this sounds! It sounds odd because we recognize that the microscope and the telescope as tools to study life and the universe. So it is with computation. Suppose that information processes already exist in nature, and the computer is our tool for studying them? David Baltimore is one of first scientists to say that computation occurs naturally; many have since followed.

UBIQUITY: So your project is really about developing a new language for discussing computing?

DENNING: Yes. That's a nice way of putting it. Over the years, we have evolved language for computing that sounds like we think we're all about programming and computing technology. I noticed that the natural sciences go to great lengths to emphasize their fundamental principles. Technologies come and go but the principles change only occasionally. About ten years ago I started to look seriously at what a principles-oriented language for computing might look like. I believed that our field was mature enough to do this, and I found many computer scientists who felt that such a project is worthwhile.

UBIQUITY: Why hasn't this been done sooner?

DENNING: I have touched on a variety of reasons already: the enrollment and offshoring issues only recently exposed the limitations of our story, and only recently has our field become mature enough to tell a different story. But even so, it has not been easy. The first few times my team and I tried to write down what we thought our principles are, we asked colleagues from all specialties each to contribute two or three principle statements for our inventory. To our surprise, our inventory reflected the traditional consensus view! Our list looked pretty much like the body of knowledge behind the ACM/IEEE curriculum. Our first attempts to express our fundamental principles wound up producing technology concept lists. But a technology concept list did not meet our goal. How could we talk with our biology and physics friends without a language and definition of computation that clearly applies to what they see happening in cell structures and quantum waves? A breakthrough happened when we saw that we had to think of computation as the principle and computer as the tool. That realization allowed us finally to construct an effective list of principles.

UBIQUITY: And you say people in the profession are generally responsive to your arguments -- that there's no particular resistance?

DENNING: I said before: recently there's been a sea change. People from other fields are saying they have discovered information processes in their deepest structures and that collaboration with computing is essential to them. We're getting better at convincing people that there is something much deeper to computing than simply programming, abstractions, chips, and networks. But this improvement is happening mostly with potential collaborators. We are still are not communicating well with prospective new computer science majors. I think our problems with enrollments in the last few years are connected with our self-image. I believe that our self-proclamations about programming have mixed combustibly with the external reputation that programming is a low-grade, easily-outsourced job, exploding the incentive for somebody to identify computing as a career to enter. Trying to continue to defend our beliefs that computing is programming and technology is taking too much energy; it is unproductive in that people outside are not buying or are not joining. To inspire interest in a career, we need to show our field as constantly engaging in big, important ideas. We want people to react: "I want to be part of that!"

UBIQUITY: Trace the timeline of your involvement for us.

DENNING: Well, my campaign, shall we call it, to communicate the fundamental principles of computing is a really a long-standing interest of mine. I first engaged with it back in 1970 when I chaired a task force as part of the NSF COSINE (computer science in engineering) project. The project was developing computing core courses that could be taught in engineering schools. My task force was charged with the question: Can we define a core course in operating systems? At that point, there was a lot of resistance to including operating systems in the core because it looked like an application area; there was only a suspicion of something deeper, emanating from the 1967 and 1969 symposia on operating system principles. My team answered affirmatively and proposed a core course on operating system principles. Within five years there were several textbooks based on the recommendation and the course was widely adopted. The course emphasized the fundamental principles behind operating systems, principles that will remain important for years to come because they cross many generations of change in computer hardware and networks. In fact, the table of contents of our report is still the table of contents of many operating system textbooks today. That was my first foray into the area of communicating what we do by looking for fundamental principles. I continued the genre in through the 1970s with six widely-used papers for *Computing Surveys*. In the middle the 1980s, I accepted an invitation from the *American Scientist* magazine to write a regular column in every issue, six times a year, on the science of computing. I purposely avoided looking at well-known fundamental issues, such as complexity of algorithms. Instead, I looked in other areas -- coordination, performance evaluation, communication, artificial intelligence, and so on, exposing fundamental principles that don't fit as algorithms and data structures. I wrote 47 columns in all, finishing in 1993. This turned out to be a very popular column and a lot of people liked learning about the deep intellectual, fundamental things that we do in computing that aren't programming. In the late 1990s I developed a course on the core ideas of information technology and my students produced a very nice [website](#) capturing everything they learned.

UBIQUITY: So would the major worry about computing in 2007 be that outsiders to the field don't appreciate its depth and range?

DENNING: I would say so. But they are beginning to appreciate it and, even more significantly, they are beginning to defend it. I was struck a couple of years ago when in some meeting someone started to criticize computer science and, before I could jump to our defense, someone else from the critic's own field gave a stout defense. He

said, we need computer science in our discipline and it's not going to go away. Several other people in the room agreed with that defender. This was the first time I heard a non-computer scientist defending the field of computer science. And then I started looking around and I was surprised at the number of collaborations that are quietly taking place, most noticeably with biology.

UBIQUITY: What are some of the other fields?

DENNING: I have mentioned that biologists see DNA as a natural code for an organism, and all the apparatus for reading and transcribing it to new living cells as natural information processing. A similar thing is happening in physics, perhaps a little more slowly; quantum physicists say that the quantum mechanics postulates information-carrying waves that manifest as physical particles and energy exchanges in the world. They have exploited this idea to produce prototypes of quantum computers and quantum cryptography, which portend a very large effort to build new technologies. I saw another example at NASA in the 1980s. The computational chemistry group was doing some pretty amazing things in using the Schrödinger wave equation, which is another quantum mechanical equation, to design new materials and new molecules. They were specifically interested in the heat shield for the Jupiter probe. The objective was to send a data-collecting satellite deep into the heavily-methane Jovian atmosphere. Their calculations with the heat shield material from the space shuttle showed that the probe would burn up only a few miles into the atmosphere. They needed a material that could withstand high-energy methane bombardment. They designed such a material by computing the methane-resistant molecule from the Schrödinger equation. With that new material, the Jupiter probe reached a great depth into the Jovian atmosphere and was very successful. Those are just three examples from traditional science where the scientists fully accepted the existence of natural information processes, exploiting them to produce some amazing new things. And of course they were very anxious to collaborate with computer scientists because computer scientists have a lot of knowledge that could help them understand their fundamental information processes.

UBIQUITY: So you're basically pretty optimistic about seeing continued and intense collaboration between computer scientists and other scientists.

DENNING: Absolutely. Even in the softer sciences like economics, social science, or management science, most everybody now accepts

that information is at the basis of what they do; information processes are part of their deep structure. They're doing some very interesting and new things by studying and exploiting those processes. So the scientific and engineering acceptance of information processes and computation is constantly widening.

UBIQUITY: In everyday life too?

DENNING: Yes, indeed. In a widely quoted article, Jeanette Wing describes how "[computational thinking](#)" has invaded almost every aspect of normal life. In New Zealand, Tim Bell and his team at the University of Canterbury have developed [Computer Science Unplugged](#), a set of methods for teaching children about computing using games, exercises, and magic tricks but no computers. The genius of their approach is that it subtly teaches that computational principles exist in life and not just on computers. There was even a political joke a few years ago: What did Bill Clinton play on his sax? Al Gore Rhythms!

UBIQUITY: What will Ubiquity readers find when they follow the link we give them, which is <http://cs.gmu.edu/cne/pjd/GP> ?

DENNING: They will find a complete description of the Great Principles of Computing Project. In addition to the project overview, readers will find a taxonomy of principles in seven categories, narrative overviews of each category, a set of top-level principles for each category, detailed expansions of each principle, an analysis of new uses of the new body of knowledge, a discussion of a Great Principles Library, links to partner projects, and answers to FAQ (frequently asked questions).

UBIQUITY: Let's quickly run down these project components. What are the seven categories?

DENNING: The seven categories are our first major contribution. They are computation, communication, coordination, recollection, automation, evaluation and design. They are groups of related principles concerning a functional area of computing. They are not technologies. No category is a principle in itself -- it's just simply a grouping of principles. I've compared the seven categories to windows in a seven-sided building that contains all computing knowledge. Each window is a distinctive way to look into the room. Something in the room can be seen from multiple windows in different ways. For example, the Internet looks like a system for moving data

(communication), a set of protocols (coordination), an information retrieval system (recollection), or a set of software layers (design).

UBIQUITY: You've distinguished the principles from practices. Why?

DENNING: The principles are statements of laws or guidelines for conduct. Practices are the actual, embodied skillful routines and habits by which we take action. You can have levels of skill in your practices, such as beginner, competent, or expert. Knowledge of the principles does not imply you can perform competently, any more than competence in action implies you can have discuss the principles intelligently. The four core practices of computing professions are programming, systems, modeling, and innovating. To be a complete computing professional, you should know the principles and be competent in the four practices.

UBIQUITY: Your site lists category narratives, top-level summaries, and full summaries. What are those?

DENNING: A full summary is a list of principles as main headings, with several subheadings under each making finer distinctions and noting implications. A top-level summary is simply the main headings extracted from the full summary. A narrative overview is a story, written at the *Computing Surveys* level, about the principle category, with less formal descriptions of the principles and more examples. A typical full summary is 9 pages long, has 6 top-level principles, and develops each principle with 5-10 detailed points.

UBIQUITY: Your site has a commentary about representing a body of knowledge. What's that?

DENNING: I noted earlier that the ACM/IEEE developed a body of knowledge summary of computing to use as a reference for deciding what to recommend for core curricula. That document lists 14 main headings and 130 subheadings. It is a technology-oriented representation of the field. ACM is expanding that representation through the Ontology Project, which aims to represent all the topics of computing with a large graph in which related topics are linked. Most of the topics in the current version of the Ontology are technology-oriented, although there is no reason not to include principles. The Great Principles framework is an alternative representation for the same knowledge space. We illustrate this with a two-dimensional matrix whose rows are technology topics and columns are the seven categories. Individual squares in the matrix contain principles relating

to the technology in the category. The GP framework covers the same space as the ACM/IEEE framework. But the GP framework emphasizes the principles, with technologies as examples. The GP framework enables new questions such as: What are all the principles of computer security? What technologies embody coordination principles? What has been the development of communication principles and what can be expected in the future?

UBIQUITY: What is the Great Principles Library?

DENNING: The GPL is a proposed addition to the ACM digital library. It would contain lots of materials to help people understand the principles and discover principle-connections among technologies. The materials would be organized into beginner, intermediate and advanced levels; beginner materials might be at the level of *Scientific American*, intermediate at the level of *Computing Surveys*, and advanced at the level of seminal or tutorial research papers. In addition to tutorial materials at the three levels there will be data sets, original sources, interviews, video clips, external links, and the like. There will be materials from within the computing field as well as from other fields that use computing principles in some way. There will be an editorial oversight process to ensure that everything in the library is carefully selected to support people's learning and understanding of principles. The overall GPL size would be limited, emphasizing the principles focus and high-quality materials. Everything in the GPL would be richly linked to other related resources such as interviews with leaders of the field or *Wikipedia* articles. A search-and-link interface would enable the library user not only to find and retrieve items, but to find multiple items sharing the same principle(s). The interface would help the user explore the seven categories and drill down to principles, tutorials, sources, and connections. Thus the library would be not only a tool for learning, but a tool for discovery of connections. Discovery of connections will support innovation and collaboration.

UBIQUITY: Once the library is established, the work is done, right?

DENNING: No. The computing field is constantly evolving and changing. As with any other representation of the field, the GP framework must also evolve and change. We believe that the rate of change in a GP framework will be slower than in a technology framework; but it will not be changeless. The GPL is a tool that helps people discover the current state of the field and how it got to be that way. The GPL is a dynamic, living entity. We're a science, and we

discover new principles all the time. Older principles fall out of use and can be retired into a “Archive” section of the library. The editorial board is responsible to keep the library up to date, to include new principles as they are discovered, to retire obsolete principles to the Archive, and to maintain the high quality and authenticity of materials. The GPL can link many articles directly from *Wikipedia* and other Web sources, after an editor confirms accuracy and quality of exposition. (As you know, *Wikipedia* is often faulted for allowing unreliable and fraudulent material.)

UBIQUITY: What do you mean by new and obsolete principles?

DENNING: A modern example is Google’s discovery of new principles about search and new ways to rank responses that people find useful. We had no such principles at the beginning of computing because we didn’t have a Web; we didn’t think about searches of that magnitude. On the other hand, design principles for a vacuum tube flip-flop, which, while still true, are nonetheless obsolete because we don’t use vacuum tubes any more. So the field itself, even described in terms of principles, may be slower changing than a technology description, but it will still be changing. A library is a useful tool to help people see what it looks like now. And an editorial process is a useful way to make sure that we have an orderly way to get new things in there and retire older principles to the Archives.

UBIQUITY: What else would our readers find on the site?

DENNING: We have links to partner projects such as Computer Science Unplugged, mentioned earlier, which illustrate some very creative and ingenious ways that people have been using the principles of computing to stir kids’ curiosity and get them excited about science and computing. Another partner is LabRats ([labrats.org](http://labrats.org)), which is a scouts-like community for getting kids excited about science.

UBIQUITY: Is that all?

DENNING: Don’t forget the Frequently Asked Questions (FAQ) page. We answer the 20 most common questions about the framework regarding principles, practices, programming, and general issues.

UBIQUITY: Besides ACM, is anyone else backing this project?

DENNING: Yes. I was just selected as one of two Distinguished Education Fellows in the recent NSF CPATH program that intends to

stimulate innovation in computing education. My project includes a summit workshop to refine the GP taxonomy and proposal for the GP library.

UBIQUITY: Any concluding remarks?

DENNING: We're hoping that many people find the GP project to be useful, not only as a language for communicating computing and discussing it among different fields, but also for a way of finding connections between technologies that use the same principles, and between fields. We think the project can help spur innovation because many innovations rest on unsuspected connections between seemingly unrelated technologies and practices. We also expect that a Great Principles Library will be an attractive venue for pioneers and innovators to write personal stories about their working with principles. Good stories inspire young people. My colleagues and I have spent a lot of time thinking about great principles, testing them in courses, interacting with lots of people, collecting ideas about what should be included, and accepting criticisms. The project is constantly evolving and improving, and it will continue to do so as we evolve it into a Great Principles Library.

UBIQUITY: That's very impressive.

DENNING: Yes, I'm beginning to hear young people say of their decisions to join computing as their major: I heard your story and I want to be part of that.

UBIQUITY: That's terrific. You've got to be proud of that.

DENNING: Yes. Computing was that way in the beginning. We all wanted to be part of it. We can reclaim that feeling now.

END

[For more on the Great Principles of Computing project, visit <http://cs.gmu.edu/cne/pjd/GP> ]