Interview of Peter J. Denning
by David Walden

[This interview was done by email in July and August 2012. This original interview is 6,400 words long, and it was edited down to about 3,300 words for publication in the October-December 2012 issue of the IEEE Annals of the History of Computing.]

David Walden, interviewee: Please tell me a bit about your early life.

Peter J. Denning, interviewee: I had interests in math, science, and nature from an early age. They were already very strong by age 10. In school, I was too small to be any good at athletics, which were socially popular. I devoted myself to my academic strengths and stayed clear of athletics. If the word "nerd" existed at the time, I would surely have been one.

By age 12 I added an interest in magician performances, especially ones that depended on mathematical tricks. By age 13 I had discovered a deep fascination with electricity and electronics, which seemed to have a magic all their own.

My parents sent me to Fairfield Prep in 1956 to get me into an intellectual community and out of the athletics-infatuated public school culture. There, under the wing of a gifted science teacher, I entered three science fairs with computers made of pinball parts and vacuum tubes -- one to compute sums, one to solve linear equations, and the last to solve cubic equations. The second computer won the science fair. That computer fasciated people because its clicking, banging, zapping, sparking, clanking step relays created all the excitements, and ozone odors, of a real pinball machine. These projectcs cemented my conviction that I could excel with the new, mysterious, emerging computing technology. The third computer worked perfectly but fared poorly at the science fair because, unlike the second maachine, it was completely silent and odorless. I paid no attention to marketing and presentation -- a life lesson.

From Fairfield Prep I went to Manhattan College Electrical Engineering in 1960. Manhattan College had not yet come into the computer age, but I got a very solid grounding in practical engineering -- the building and testing of things that other people could use.

I came out on top of my class at Manhattan in 1964 and got an NSF fellowship good at any graduate school. I applied to MIT in fulfillment of my father's advice. Many faculty at Manhattan warned against MIT because no prior Manhattan graduate had passed the PhD exams there. But I went anyway because my savvy dean told me that it would be tough but I could do it.

DW: Say a bit about MIT (I know you taught course 6.253 because I was in your class).

PJD: MIT was indeed tough. I had to relearn all my EE because MIT had a completely different philosophy from Manhattan College about the principles and organization of EE. I took all the MIT EE core courses plus my required master's courses in my first year. That intense preparation was barely enough. It took me two tries to pass the PhD qualifiers, and my master's thesis advisor, Jack Dennis, took me under his wing. He and I have had a long and productive relationship and friendship for almost fifty years now.

My MS thesis was about scheduling requests for a rotating disk or drum memory so as to minimize mean access time. Access time to secondary storage was a critical issue for speed in the time sharing system Jack Dennis was developing at the time. During that year I worked closely with Allan Scherr who taught me about systems programming, language design, compiling, data collection in an operating system kernel, discrete simulation, and queueing theory -- all things I used in my master's thesis. Through the thesis, Jack and I showed that shortest latency time disk scheduling was optimal for time sharing systems.

For my PhD research, I decided to tackle a much tougher resource allocation problem, which was looming in the design of Multics. The problem was now to measure the processor and memory demands of multi-threaded computations, configure a system with appropriate capacity for the demand, and manage the allocation of CPU and memory dynamically. The most immediate problem was the Multics virtual memory. They knew how to structure it but not how to build measurement and control systems to maintain stable system operation. Jerry Saltzer told me of thrashing, a problem they were encountering with multiprogrammed virtual memory systems, and challenged me to find a solution. That solution turned out to be much harder than either of us imagined. My quest produced the theory of locality, the working set model for program behavior, and a method of system balance for optimal control.

During the PhD years I also helped Jack Dennis teach a course 6.253 on computational models. I loved teaching that material and, through the teaching, developed a deep understanding of computation and the essential role of machines in doing it. Our class notes caught the attention of the Prentice-Hall editor. We signed a contract for a book, Machines, Languages, and Computation, in 1967. Unfortunately, writing a book was more work than I ever imagined, and it took us a decade to finally finish it in 1978.

In January 1968 Jack told me I had plenty of material for my PhD thesis through my published papers and technical reports. He gave me until May to finish up. I went on a crash program of writing and working with my committee. I graduated with my MIT PhD in May 1968. Unfortunately, I could not share the good news with that savvy dean at Manhattan, for he had succumbed to an automobile accident.

As graduation approached, I struggled with the question of where to go next. I had offers from MIT, Stanford, Cornell, and Princeton. I befriended Lotfi Zadeh during a visit and he persuaded me that I would enjoy better professional development by leaving MIT and being in a new culture. I chose Princeton because it was most attractive to my family.

DW: My memory is that at Princeton you continued and expanded the scope of your research in the areas of operating systems, applying queuing theory, etc. Presumably you also did some teaching and supervision of graduate students, and in general worked to develop your career in academia. Please tell me a bit about that.

PJD: My Princeton years, 1968-1972, were productive. I developed and taught new courses in principles of operating systems and computer architecture. I took on two PhD students and helped with several others. I collaborated closely on several projects with computer scientists Ed Coffman, Jeff Ullman, and Al Aho, and with electrical engineers Stuart Schwartz and Bruce Eisenstein. Those projects extended the working set theory and validated it with experiments, and also codified operating systems principles. I published twenty research papers on those topics.

The year before I went to Princeton, I had helped Jack Dennis organize the first ACM symposium on operating system principles (SOSP), held in Gatlinburg, TN, in 1967. That symposium uncovered a huge interest in the fundamental principles of operating systems, which were the most complex computing systems then known. At Princeton, Ed Coffman and I organized a follow-on, SOSP-2, in 1969. The SOSP has continued every two years since that time.

In 1969 and 1970 I was invited to chair a task force for an NSF project called COSINE, computer science in engineering, which was developing prototypes of new core courses for computer science programs. I invited Jack Dennis, Nico Habermann, Butler Lampson, and Dennis Tsichritzis to the team. Our mission was to propose a core course on operating systems principles. This was challenging and fun and meshed nicely with my interests in the SOSP. Our recommendations, released in 1971, were soon adopted nationally as many universities created their first systems oriented core courses based on our recommendations. The course outline we provided at the time can still be seen as a subset of the topics in modern operating systems textbooks.

Immediately after I finished with the task force, Ed Coffman and I decided to write a book with the bold title OPERATING SYSTEMS THEORY. We put it on a fast track and published the book in 1973. A few years later Prentice-Hall asked if we would do a second edition. There had been so many new developments in operating systems we thought we would actually have to write a new book. So, we declined the opportunity for a second edition. That was not a bad choice because the book stayed in print until around 1995.

DW: I believe you were recruited to Purdue. Please tell me about your decision to move there, and how things worked out.

PJD: In my fourth year at Princeton, I concluded that getting promoted at Princeton did not look like a good option. Because of a cap on tenured faculty, there would be no more than two promotions in the engineering school in the next five years, and at most one in the EE department. The CS faculty were a minority in the EE department and the traditional EE's already had a candidate they wanted to promote. Early in 1972 I encountered Sam Conte, the chair of CS at Purdue, on an elevator at a conference. Sam said: "I hear you are not so happy at Princeton. Please consider us. I can make you an offer as tenured associate professor and pay you 50 percent more salary." Now that was a great elevator pitch!

I interviewed at Purdue soon thereafter in the dead of winter. The faculty were incredibly warm and welcoming. I departed with Sam's offer in hand and accepted it a couple of weeks later. Ed Coffman left Princeton about the same time for similar reasons. He used to joke about Princeton's inability to hold on to its computer scientists: "Princeton is a great place to have been from."

My Purdue years were also very productive. With the help of several graduate students I completed the working set project in 1980. My colleagues, students, and I showed that the working set model could simulate any paging algorithm whose memory contents obeyed an inclusion property with increasing value of the control parameter. We validated the phase transition model of locality and used it to show that working set memory management was within 5% of optimal for memory controllers that could not see the future. We had answered all the basic questions about locality and memory system management we set out in 1966. I published my final research paper on working sets in 1980.

I see that today there are aspects of the working set issue around which understanding has been lost. For example, many consider the working set to be an unrealizable ideal, even though we showed a simple method of implementing it that was no more expensive than other methods used in operating systems. Linux thrashing problems could be prevented by a simple switch to working set detection.

By 1980, I saw new opportunities in system performance modeling, which used queueing theory, my favorite branch of mathematics. I entered a productive collaboration with Jeff Buzen, who had made queueing network models practical in 1971 with his fast algorithm for computing throughput and response time. He was interested in reformulating queueing theory around assumptions commonly met by computing systems. I helped him develop the theory he called "operational analysis" and showed that this theory applied to queueing network models. We showed our work to colleagues in the Industrial Engineering Department, who immediately applied it to factory production lines. During the Purdue years I published many more papers and another book, and advised ten PhD students. (Three of them went on to receive "distinguished alumni" awards from Purdue.) I was promoted to full professor in 1975 and became head of department in 1979, replacing Sam Conte, who wanted to retire.

As you can see I had a pretty conventional academic career through the 1970s. When I became department head, that began to change.

DW: How did you meet Dorothy?

PJD. I met her in late 1971 when I visited University of Rochester to interview for chair of their new CS department. She was an instructor. I saw her again at the 1972 Spring Joint Computer Conference. When she described her interest in getting a PhD, I suggested she apply immediately to Purdue. Sam Conte was impressed with her credentials and hired her as instructor. Our relationship soon changed because we fell in love and got married in early 1974. She became renowned in data security and cryptography. We have had a long and happy marriage.

DW: My memory is that already at MIT you were beginning to be active in professional society activities, including editing and great concern for good writing. [In fact, while visiting you in your office at Tech Square, I'm pretty sure I remember seeing you reading Strunk and White's little style book. (I am more sure I remember seeing you reading Cox and Smith's little book on queues and using some little sorting system with with little rods and notches cut our of stiff cards.)] And you are well known for practically a second career with the ACM. What motivated all that?

PJD: You ask two separate, but related questions. Let me address the writing one first. From a young age I displayed a knack for writing and liked doing it. My parents told me that my letters from summer camp were captivating. In high school, my science teacher told me I had a knack for exposition and expressing difficult ideas clearly, and he gave me my first teaching experience that included writing a series of lectures about basic electricity for the science club. I also wrote articles and even drew cartoons for the school's magazine. My girlfriends said that my letters to them were always interesting. In college I won a couple of awards for essays I wrote. At MIT I wrote extensive notes for the course 6.253, as mentioned above, and was invited to join with Jack Dennis on a book. In 1966, Jerry Saltzer completed his PhD thesis, which I admired as a masterpiece of clear exposition. He had the first definition of "process" that I actually understood -- he said a process is a program in execution on a virtual machine. (At the time, operating system people were still trying to settle on what they meant by a process.) Jerry told me that he valued clear writing and used Strunk and White as a guide. I got a copy, loved it, and used it as my guide too. Later I found a more advanced guide on English usage by H. W. Fowler; it is still available as a classic. Two of the papers I wrote at Princeton received best paper awards. My love of writing and exposition had well prepared me for an active publication and editing sub-career.

Your other question is about ACM. I learned about ACM from MIT faculty and joined as a student member in 1965. When I attended the SOSP in 1967, Walter Kosinski, the co-organizer with Jack Dennis, suggested that I could broaden my interests and link up with more people in operating systems by joining the Special Interest Committee on Time Sharing (SICTIME). The principals of SICTIME not only welcomed me as a member, but they recruited me to be the SICTIME newsletter editor. I was honored to be able to contribute my writing talents and meet new operating systems people at the same time. I faithfully wrote and distributed SICTIME newsletters for the next two years.

While I was editor, I joined discussions with SICTIME leaders about the possibility of transforming SICTIME to the higher-level status of an ACM special interest group. I advocated that as part of the transition, they should rename to focus on operating systems rather than time-sharing. They took my suggestion and put me in charge of writing up the transition plan and the new bylaws. We got approval from the ACM Council in 1969, and soon thereafter president Bernie Galler appointed me as chair of the new SIGOPS.

In early 1970 the ACM Council approved a bylaw change that created a SIG board to oversee the 32 SIGs and added its chair to Council. I got a call from the nominating committee that spring asking if I would be a candidate for SIG Board chair. I was flattered but very reluctant because I did not want to jeopardize my chances at promotion at Princeton. At the same time, I was attracted to the opportunity to help ACM get the fractious SIGs organized and on better terms with the ACM Council. I decided I would run, but reckoned I would not be elected since the other candidate, Patrick Fischer, was an old and respected hand in the SIGs. I was certain that Patrick would win. Imagine my surprise when the ACM secretary called up to tell me I'd won the SIG Board election and was now a member of Council.

That was a turning point in my career. I learned how to arrange my daily schedules so that I would have time for both my professorial work and my ACM work. It's possible I could have been more productive with academics and research, but even with the weight of a "second career" I was still more productive academically than most of my colleagues at Princeton and Purdue, which was good enough to convince them I was still a genuine academic.

In your question you also mentioned my interest in queues and in edge-notched cards. In 1966, Allan Scherr got me interested in queues because of his success in using the Machine Repairman queueing model to predict the CTSS performance. When you saw me reading Cox and Smith, I was in a period of voracious reading about queueing theory. I wanted to use the theory to help me with the problems of resource allocation in multi-process computer systems in my PhD research.

The edge-notched cards were an interesting technology. It was invented many years before, by a gentleman named McBee around 1896 I understand. The cards were all the same and had holes punched along the edges. Each hole represented an attribute of the data written on the card. With a special tool you could snip out the paper between the hole and the edge; the notch indicated that the data had the attribute associated with the hole. You could then pass a rod through one of the holes on a deck of cards and lift it up. The cards with the attribute (notched) would stay behind. You could do multiple attributes with multiple rods. It was a clever sorting system. IBM card sorters superseded McBee cards in the 1930s, but was still around as a "quaint technology" in the 1970s.

DW: I read your articles in the American Scientist for a number of years. How did that come about?

PJD: The American Scientist project began in 1985 when I was Director of RIACS (Research Institute for Advanced Computer Science) at NASA-Ames. The AmSci editor, Michelle Press, had seen my columns in 1980-82 while I was ACM President, and some of my other writings on behalf of computational science, and thought I could do a column for them on "The Science of Computing". The objective of the column was to reveal the science side of computing so that scientists in all the fields served by AmSci could come to appreciate computer science. In those days, few people outside computing understood computer science and many thought it was a guild of programmers. The idea that computing is science was novel. Of course I found this very appealing and immediately agreed to do it. From 1985-1993 I wrote 47 columns for the magazine. I ended my gig after Sigma Xi, the parent organization of AmSci, decided to move its headquarters from New Haven to Raleigh amd Michell Press and all her staff quit rather than move. One of my undone projects is to scan those columns and put them up on my website -- the science is unchanged!

DW: I do hope you soon post those on your website.

Despite a successful tenure at Purdue, including becoming department head, you left and went to RIACS and from there to George Mason. It seems like you were seeking something. What was that? In particular, by the time you were at George Mason, you seemed deeply into the management thinking of Fernando Flores.

PJD: Indeed, a lot of dots to connect! The best way to understand my transitions is to understand the quests I was on. My lifelong professional quest has been to understand the fundamental principles of computing. This has been driven by my ongoing sense of the magic, beauty, and joy of computing, and my ongoing conviction that computing can make big contributions across the board in the physical, life, and social sciences. When computing was criticized for not being a true science, I brought forth evidence that computing is science. When computing was characterized as programming, I brought forth evidence that computing is much broader and programming is a core practice. When computing was characterized as abstractions, I brought forth evidence that the system and machine architecture is critical. This quest remains central to me as you can see in the great principles of computing project (greatprinciples.org).

Another quest, which developed in the 1980s, is to help computer scientists successfully contribute their computing expertise to people in other fields seeking to solve problems. I discovered that the culture of my own education accentuated my childhood nerd tendency, which works against reaching out. I saw that many of my students had their own aspirations to discover new ideas and get them adopted, and were encountering the same problems as I did in finding ways to engage with people in other domains. I wanted to help. I did not like the caricature that computer scientists were anti-social and I wanted to help my students overcome that image. This quest led me to study management and leadership with Fernando Flores and Richard Strozzi-Heckler. It led me to learn how to coach students in being more effective at getting their ideas adopted. It led me to propose reforms in engineering education that would help all students be good designers and contributors. It finally led me to a novel understanding of innovation process, which my co-author Bob Dunham and I made the subject of our book ("The Innovator's Way", 2010).

These general threads led me into a number of sub-quests. Dorothy and I wound up moving to new locations and organizations about every ten years in order to immerse in environments where we could most effectively pursue those quests.

Now I can comment on the transitions you noted. Let's begin with moving from Purdue in Indiana to RIACS in California. Soon after I become department head in 1979 I discovered that I was ineffective at getting the other department heads in my School of Science to think of CS as a science rather than as a technology. At first I thought the problem was my communication skills. I spent several years improving them, but it helped only marginally. I reluctantly concluded that most of the problems we computer scientists were working on -- such as faster computers or better operating systems -- did not strike those outside our field as issues of intellectual substance. Cryptography and robotics were among a handful of areas that drew genuine interest. I wanted to get into a different environment and learn how to make computer science more interesting. NASA-Ames was perfect. Besides, Dorothy and I had been dreaming of getting to California. We moved in 1983.

Our mission at RIACS was to help NASA-Ames research groups move forward through computational science, a combination of computation and computer science. I hired computer science researchers and matched them up with NASA researchers in high performance computation, artificial intelligence, and networking. We were quite successful. Yet we still had persistent problems with customer relations. For example, our scientists were pretty sloppy about meeting deadlines for deliverables; they thought research results could not be scheduled. This was not good in a research environment funded only when the customers are satisfied. I saw this as an area where I needed some more professional development. Terry Winograd introduced me to his colleague Fernando Flores, who had a new theory of management based on communication rather than decision making. (Winograd and Flores wrote a hard-hitting critique of AI and cognition that appealed greatly to the NASA AI people.) I studied management and leadership with Flores. I began to apply his philosophy to my work at RIACS. I was amazed at how much more productive we became when we got his ideas into practice, and how much more trust we earned from our customers. I was also amazed at how much I had to unlearn. Some of those old nerd habits lingered and were hard to break.

CSNET was a project that overlapped Purdue and RIACS. In 1979 I joined up with Larry Landweber, Dave Farber, and Tony Hearn to propose a computer science network that would make the ARPANET technologies available to all CS researchers and their students. At the time, ARPANET was closed to universities except for a few who had ARPA contracts. The network-connected universities were rapidly moving ahead of the others, and we did not want to be left behind. We got the CS community behind the CSNET proposal and NSF funded it for five years. By 1986, CSNET included 120 CS departments and industry labs, about 50,000 students, and was self-supporting. CSNET received the 2009 Postel Award from the Internet Society for its pivotal role in transitioning from the ARPANET to the NSFNET.

By 1990 I had concluded that the customer-relations issues I had grappled with were not peculiar to RIACS. Many computer scientists did not know how to reach out to people in other fields to contribute computing expertise to solve their problems. My colleagues liked their identity as programmers even as they chafed when other people only wanted them to program rather that design new solutions to problems. It seemed to me that this revealed a defect in the way we educated computer scientists. I wrote a manifesto about needed reforms in computing education, which drew a lot of attention and caused a stir. I needed to return to academia to pursue this agenda. Dorothy was also anxious to return to academia. We found two positions in the Washington, DC, area in 1991 -- she became CS department chair at Georgetown and I did the same at George Mason.

At George Mason I linked up with other reform-minded faculty. We encountered a lot of resistance from faculty who liked the principles behind our proposed project-oriented and customer-oriented reforms but did not want to change their practice. We found many students who longed the reforms we were advocating. I concluded that I still needed more professional development in leadership to help interested faculty to change and help students learn how to get their ideas adopted. I undertook further leadership studies with Richard Strozzi Heckler in my spare time. I learned how to coach students as was certified as a Master Somatic Coach. I initiated a design course called Sense 21 that aimed to put Flores-Heckler principles to work to help students be more successful as designers and get their ideas adopted as innovators. Sense 21 was a big success. Its graduates formed an alumni club so that they could continue learning together. That group lived on until I returned to California in 2002. I never had any graduates of my operating systems classes want to form an alumni group!

Also at George Mason I continued work on my quest to understand the fundamental principles of computing. I designed and implemented a new capstone course on the core of information technology.

Dorothy and I wanted someday to return to California. Suddenly in 2002 an opportunity to do so appeared when two professor positions opened at Naval Postgraduate School in Monterey. We took them and returned.

At NPS I focused on two main projects: great principles of computing and innovation. My colleagues and I wrote down the fundamental principles of computing. We gathered evidence demonstrating that computer science is not only a science of the artificial, it is also a natural science. We designed a course on the great principles that became one of our department's best courses. In 2004, our faculty reorganized the curriculum using the principles framework as a guide. Our website greatprinciples.org has all the materials.

My other main project was the innovation project. After all my years of study around leadership of research, teaching students some of the key leadership practices in Sense 21, and working with my colleague Bob Dunham to teach leadership practices to business people, I realized I had learned some new things worth sharing innovation. Bob and I collaborated on THE INNOVATOR'S WAY (MIT Press 2010), which is about eight essential practices for successful innovation. This work received good reviews and generated a lot of interest.

DW: You didn't mention ACM, which seems to have been a "second career" for you. What role did ACM play in the quests you discussed above?

PD: Computer science has always been my first and foremost career. ACM became a sort of second career that never eclipsed the first, providing many rich channels to pursue my quests. My pursuit of innovation practice in the last two decades has become a third career.

In ACM I exercised leadership roles continuously since 1970, when I was elected as the first SIG Board chair. Here is a list of all my major ACM positions:

- SIG Board chair (1970-74)
- Editor in Chief, Computing Surveys (1976-78)
- Council member-at-large (1974-78)
- Vice president (1978-80)
- President (1980-82)
- Editor in chief of ACM Communications (1983-92)
- Chair Publication Board and head of digital library project (1992-98)
- Chair of Education Board (1998-2004)
- Director ACM IT Profession Initiative (1999-2002)
- Member, ACM Education Council (2004-present)
- Editor in Chief, ACM Ubiquity (2008-present)

One of the big accomplishments when I was president was a plan for restructuring publications, which were a major source of complaint for many members. In 1983 I became the editor-in-chief of the Communications with a mission to transform it from a narrow, sleepy research journal into a lively magazine for all members. I retired from that in 1992, when I was elected chair of the Publications Board, which was newly charged with developing and implementing an ACM Digital Library. I led the team that developed and tested prototypes until converging on an architecture that would work. We developed a wholly new set of copyright policies that anticipated the Creative Commons by seven years. We developed a business plan that paid for the transition from existing funds and was priced reasonably from a member standpoint. The digital library evolved into ACM's "crown jewel" since we rolled it out in 1997.

In 1998, with the DL project behind me, I chaired the ACM Education Board, where we initiated a new curriculum recommendation in 2001 and a new professional development online center. I also led the "IT Profession Initiative" with the ACM Council to have ACM become the society for computing professionals. I retired from the Ed Board and the IT initiative in 2004 and remained a member of the Education Council ever since. I also took on the Editor-in-Chief of ACM Ubiquity, and online peer reviewed magazine devoted to the future of computing and the people creating it.

All these activities over the years served my quests for gaining recognition of computing as a full science and finding ways to encourage computing people to reach out to other professions. I am happy to see that many other people saw merit in the same goals. We now have a vibrant profession that is seen as much broader than programming and excites a sense of magic and beauty in young people. Information is so pervasive that some people, myself included, now perceive computing as the fourth great domain of science, alongside the physical, life, and social sciences.

DW: When you have done something more technical such as developing the working set model or collaborating in the development of CSNET, the impact is relatively clear -- the model was widely applied and the network came into existence and was widely used. Your work for better recognition of the profession or the science seems less technical. How can you be sure of the impact -- are these ideas in fact being widely adopted and applied?

PD: I do many things without calculating whether, in time, there will be a big impact. I am more concerned about whether I am bringing value to someone right now.

My work has long been a mixture of technical and professional. After I finished working on memory management and locality, I worked on queueing networks, which are powerful models of computer systems. The "operational approach" Jeff Buzen and I worked out is now commonly used in performance evaluation textbooks.

CSNET mixed technical accomplishment (new protocols and networks) with community building. It was cited in 2009 by the Internet Society for being a critical step in the transition from old ARPANET to the NSFNET, which was the backbone of the modern Internet. At the time, I was not concerned with making history -- only about building a community.

My work on the great principles of computing is deeply technical. I have had to dig deeply into the details of every corner of the field in order to understand the simplest ways to express key principles. Even something seemingly simple, like what is information, turns out to be plagued with paradoxes. For example, we say that computers process information without regard to its meaning, and yet users find meaning in computer outputs all the time. So is meaning irrelevant or is it the purpose of computing? Information is at the heart of computing. I am still struggling to understand what information is and how computing generates new information.

I took on the CACM editorship in 1983 because I wanted to help the CACM become a channel to further everyone's understanding of computing. I think we succeeded. CACM won a number of awards and helped draw more members to ACM. Then I took on the ACM DL project in 1992 because I wanted to help ACM establish an electronic community and expand the number of people who had access to CS knowledge. That succeeded too; the DL is now seen by ACM as its "crown jewel".

In teaching, I have always found a reward in a student's joy of discovering a truth for themselves. At George Mason, I organized the course Sense 21 not because I had an aspiration to change the way engineers approach design, but because I knew some things about language and action that could help my students be more successful. I felt compassion for students who felt overwhelmed, distrusted, or unable to get their ideas adopted. By teaching them about action, emotions, and moods in language, I helped them with these problems. What a reward it was at the end of the course when one student after another said that the course was a life-changing experience for them. And when they wanted to perpetuate the learning group by forming an alumni society!

Although it was not my goal to have a larger impact, these teaching concerns wound up having an impact larger than I imagined. I learned this was so when I was recognized by George Mason and the Commonwealth of Virginia with three teaching awards.

When I undertook the columns for American Scientist, I was on a quest to help skeptical scientists in other fields come to appreciate computer scientists. I knew we were not seen as peers because NASA people looked at us as programmers, not as fellow scientists. My colleagues expressed their appreciation for this work in the citation for the CRA distinguished service award.

Today, a lot of people have joined me in my interest to understand the fundamental principles of computing. The Educational Testing Service is now designing new Advanced Placement course on CS Principles.

In 2009 I organized a conference called Rebooting Computing: The Magic and Beauty of Computer Science. I did that because may people in the field were in anguish over the lack of acceptance of computer scientists as peers in other sciences, and the decline of students choosing CS as their major in college. The purpose was to stimulate conversations between segments of the field that had little to say to each other, and in so doing to recover the sense of magic and beauty that had brought so many others into the field. The summit was successful. It is now common to see K12 teachers and university professors working together on the educational pipeline, to hear the words magic, joy, and beauty in student conversations about what they are learning, and to find these words in the titles of new university courses.

I have always been drawn to Lao Tzu's statement, "A leader is best when people barely know he exists. Of a good leader, who talks little, when his work is done, his aim fulfilled, they will say, 'We did this ourselves.'"

DW: You have a perhaps uniquely wide career of important contributions in many directions. Do you sometimes wish you had focused more on developing innovative technology?

PD: No, I do not regret my decisions to branch out from pure technology development. I would not have been able to make contributions to teaching, image of our field, and building of our communities. I believe that each of my contributions has had an impact, even though at the time making an impact was not my concern. I believe that some of my work has helped others build technology that has impact.

I have always had a concern for doing an excellent job at whatever I do. I don't always succeed. But I am able to look back at most of my work and see that it was good.