2009-12

The Profession of IT, Computing's Paradigm

Denning, Peter J.

Computing's Paradigm (with Peter Freeman) (December 2009) Trying to categorize computing as engineering, science, or math is fruitless; we have our own paradigm.

DOI:10.1145/1610252.1610265

Peter J. Denning and Peter A. Freeman

The Profession of IT
Computing's Paradigm

Trying to categorize computing as engineering, science, or math is fruitless; we have our own paradigm.

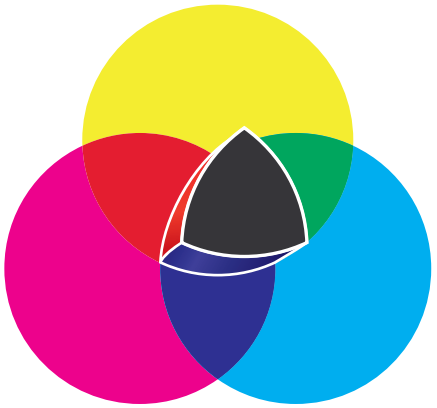
COMPUTING RIGHTFULLY COMES up in many discussions of university organization and curricula, high school courses, job qualifications, research funding, innovation, public policy, and the future of education. In repeated attempts to characterize our field in these discussions, our leaders continue to encounter sometimes contentious debate over whether computing is a field of engineering or science. Because it leaves others with a sense that we lack a clear focus, that debate negatively influences policies involving computing.

There seems to be agreement that computing *exemplifies* engineering and science, and that neither engineering nor science *characterizes* computing. What then does characterize computing? In this column, we will discuss computing's unique paradigm and offer it as a way to leave the debilitating debate behind.

The word "paradigm" for our purposes means a belief system and its associated practices, defining how a field sees the world and approaches the solutions of problems. This is the sense that Thomas Kuhn used in his famous book, *The Structure of Scientific Revolutions*. Paradigms can contain sub-paradigms: thus, engineering divides into electrical, mechanical, chemical, civil; science divides into physical, life, and social sciences, which further divide into separate fields of science.

Roots of the Debate

Whether computing is engineering or science is a debate as old as the field



itself. Some founders thought the new field a branch of science, others engineering. Because of the sheer challenge of building reliable computers, networks, and complex software, the engineering view dominated for four decades. In the mid-1980s, the science view began to assert itself again with the computational science movement, which claimed computation as a new sub-paradigm of science, and stimulated more experimental research in computing.

Along the way, there were three waves of attempts to provide a unified view. The first wave was by Alan Perlis,[9] Allen Newell,[8] and Herb Simon,[11] who argued that computing was unique among all sciences and engineering in its study of information processes. Simon went so far as to call computing a science of the artificial.

The second wave started in the late 1960s. It focused on programming, seen as the art of designing information processes. Edsger Dijkstra and Donald Knuth took strong stands favoring programming as the unifying theme. In recent times, this view has foundered because the field has expanded and the public understanding of programmer has become so narrow (a coder).

The third wave was the NSF-sponsored Computer Science and Engineering Research Study (COSERS), led by Bruce Arden in the mid-1970s. It defined computing as automation of information processes in engineering, science, and business. It produced a wonderful report that explained many exotic aspects of computing to the layperson.[1] However, it did not succeed in reconciling the engineering and science views of computing.

Peaceful Coexistence

In the mid-1980s, the ACM Education Board was concerned about the lack of a common definition of the field. The Board charged a task force to investigate; its response was a report *Computing as a Discipline*.[4] The central argument of the report was that the computing field was a unique combination of the traditional paradigms of math, science, and engineering (see Table 1). Although all three had made substantial contributions to the field, no single one told the whole story. Programming—a practice that crossed all three paradigms—was essential but did not fully portray the depth and richness of the field.

The report in effect argued for the peaceful coexistence of the engineering, science, and math paradigms. It found a strong core of knowledge that supports all three paradigms. It called on everyone to accept the three and not

try to make one of them more important than the others.

Around 1997, many of us began to think the popular label IT (information technology) would reconcile these three parts under a single umbrella unique to computing.[3,7] Time has proved us wrong. IT now connotes technological infrastructure and its financial and commercial applications, but not the core technical aspects of computing.

A Computing Paradigm

There is something unsatisfying about thinking of computing as a "blend of three sub-paradigms." What new paradigm does the blend produce?

Recent thinking about this question has produced new insights that, taken together, reveal a computing paradigm. A hallmark of this thinking has been to shift attention from computing machines to information processes, including natural information processes such as DNA transcription.[2,6] The great principles framework interprets computing through the seven dimensions of computation, communication, coordination, recollection, automation, evaluation, and design (see http://greatprinciples.org). The relationships framework interprets computing as a dynamic field of many "implementation" and "influencing" interactions.[10] There is now a strong argument that computing is a fourth great domain of science alongside the physical, life, and social sciences.[5]

These newer frameworks all recognize that the computing field has expanded dramatically in the past decade. Computing is no longer just about algorithms, data structures, numerical methods, programming languages, operating systems, networks, databases, graphics, artificial intelligence, and software engineering, as it was prior to 1989. It now also includes exciting new subjects including Internet, Web science, mobile computing, cyberspace protection, user interface design, and information visualization. The resulting commercial applications have spawned new research challenges in social networking, endlessly evolving computation, music, video, digital photography, vision, massive multiplayer online games, user-generated content, and much more.

There is an interesting distinction between computational expressions and the normal language of engineering, science, and mathematics.

The newer frameworks also recognize the growing use of the scientific (experimental) method to understand computations. Heuristic algorithms, distributed data, fused data, digital forensics, distributed networks, social networks, and automated robotic systems, to name a few, are often too complex for mathematical analysis but yield to the scientific method. These scientific approaches reveal that *discovery* is as important as *construction* or *design*. Discovery and design are closely linked: the behavior of many large designed systems (such as the Web) is discovered by observation; we design simulations to imitate discovered information processes. Moreover, computing has developed search tools that are helping make scientific discoveries in many fields.

The newer frameworks also recognize natural information processes in many fields including sensing and cognition in living beings, thought processes, social interactions, economics, DNA transcription, immune systems, and quantum systems. Computing concepts enable new discoveries and understandings of these natural processes.

The central focus of the computing paradigm can be summarized as

Table 1. Sub-paradigms embedded in computing.

	Math	Science	Engineering
Initiation	Characterize objects of study (definition)	Observe a possible recurrence or pattern of phenomena (hypothesis)	Create statements about desired system actions and responses (requirements)
Conceptualization	Hypothesize possible relationships among objects (theorem)	Construct a model that explains the observation and enables predictions (model)	Create formal statements of system functions and interactions (specifications)
Realization	Deduce which relationships are true (proof)	Perform experiments and collect data (validate)	Design and implement prototypes (design)
Evaluation	Interpret results	Interpret results	Test the prototypes
Action	Act on results (apply)	Act on results (predict)	Act on results (build)

Table 2. The computing paradigm.

	Computing
Initiation	Determine if the system to be built (or observed) can be represented by information processes, either finite (terminating) or infinite (continuing interactive).
Conceptualization	Design (or discover) a computational model (for example, an algorithm or a set of computational agents) that generates the system's behaviors.
Realization	Implement designed processes in a medium capable of executing its instructions. Design simulations and models of discovered processes. Observe behaviors of information processes.
Evaluation	Test the implementation for logical correctness, consistency with hypotheses, performance constraints, and meeting original goals. Evolve the realization as needed.
Action	Put the results to action in the world. Monitor for continued evaluation.

information processes—natural or constructed processes that transform information. They can be discrete or continuous.

Computing represents information processes as "expressions that do work." An expression is a description of the steps of a process in the form of an (often large) accumulation of instructions. Expressions can be artifacts, such as programs designed and created by people, or descriptions of natural occurrences, such as DNA and DNA transcription in biology. Expressions are not only representational, they are *generative*: they create actions when interpreted (executed) by appropriate machines.

Since expressions are not directly constrained by natural laws, we have evolved various methods that enable us to have confidence that the behaviors generated do useful work and do not create unwanted side effects. Some of these methods rely on formal mathematics to prove that the actions generated by an expression meet specifications. Many more rely on experiments to validate hypotheses about the behavior of actions and discover the limits of their reliable operation.

Table 2 summarizes the computing paradigm with this focus. While it contains echoes of engineering, science, and mathematics, it is distinctively different because of its central focus on information processes.[5] It allows engineering and science to be present together without having to choose.

There is an interesting distinction between computational expressions and the normal language of engineering, science, and mathematics. Engineers, scientists, and mathematicians endeavor to position themselves as outside observers of the objects or systems they build or study. Outside observers are purely representational. Thus, traditional blueprints, scientific models, and mathematical models are not executable. (However, when combined with computational systems, they give automatic fabricators, simulators of models, and mathematical software libraries.) Computational expressions are not constrained to be outside the systems they represent. The possibility of self-reference makes for very powerful computational schemes based on recursive designs and executions, and also for very powerful limitations on computing, such as the noncomputability of halting problems. Self-reference is common in natural information processes; the cell, for example, contains its own blueprint.

The interpretation "computational thinking"[12] embeds nicely into this paradigm. The paradigm describes not only a way of thinking, but a system of practice.

Conclusion

The distinctions discussed here offer a distinctive and coherent higher-level description of what we do, permitting us to better understand and improve our work and better interact with people in other fields. The engineering-science debates present a confusing picture that adversely affects policies on innovation, science, and technology, the flow of funds into various fields for education and research, the public perception of computing, and the choices young people make about careers.

We are well aware that the computing paradigm statement needs to be discussed widely. We offer this as an opening statement in a very important and much needed discussion. C

References

1. Arden, B.W. *What Can Be Automated: Computer Science and Engineering Research Study (COSERS)*. MIT Press, 1983.
2. Denning, P. Computing is a natural science. *Commun. ACM 50*, 7 (July 2007), 15–18.
3. Denning, P. Who are we? *Commun. ACM 44*, 2 (Feb. 2001), 15–19.
4. Denning, P. et al. Computing as a discipline. *Commun. ACM 32*, 1 (Jan. 1989), 9–23.
5. Denning, P. and P.S. Rosenbloom. Computing: The fourth great domain of science. *Commun. ACM 52*, 9 (Sept. 2009), 27–29.
6. Freeman, P. Public talk "IT Trends: Impact, Expansion, Opportunity," 4th frame; www.cc.gatech.edu/staff/f/freeman/Thessaloniki
7. Freeman, P. and Aspray, W. *The Supply of Information Technology Workers in the United States*. Computing Research Association, 1999.
8. Newell, A., Perlis, A.J., and Simon, H.A. Computer science, letter in *Science 157*, 3795 (Sept. 1967), 1373–1374.
9. Perlis, A.J. The computer in the university. In *Computers and the World of the Future*, M. Greenberger, Ed. MIT Press, 1962, 180–219.
10. Rosenbloom, P.S. A new framework for computer science and engineering. *IEEE Computer* (Nov. 2004), 31–36.
11. Simon, H. *The Sciences of the Artificial*. MIT Press (1st ed. 1969, 3rd ed. 1996).
12. Wing, J. Computational thinking. *Commun. ACM 49*, 3 (Mar. 2006), 33–35.

Peter J. Denning (pjd@nps.edu) is the director of the Cebrowski Institute for Information Innovation and Superiority at the Naval Postgraduate School in Monterey, CA, and is a past president of ACM.

Peter A. Freeman (peter.freeman@mindspring.com) is Emeritus Founding Dean and Professor at Georgia Tech and Former Assistant Director of NSF for CISE.