The History of Computing in the History of Technology

Michael S. Mahoney
Program in History of Science
Princeton University, Princeton, NJ

(*Annals of the History of Computing* 10(1988), 113-125)

After surveying the current state of the literature in the history of computing, this paper discusses some of the major issues addressed by recent work in the history of technology. It suggests aspects of the development of computing which are pertinent to those issues and hence for which that recent work could provide models of historical analysis. As a new scientific technology with unique features, computing in turn can provide new perspectives on the history of technology.

Introduction

Since World War II 'information' has emerged as a fundamental scientific and technological concept applied to phenomena ranging from black holes to DNA, from the organization of cells to the processes of human thought, and from the management of corporations to the allocation of global resources. In addition to reshaping established disciplines, it has stimulated the formation of a panoply of new subjects and areas of inquiry concerned with its structure and its role in nature and society (Machlup and Mansfeld 1983). Theories based on the concept of 'information' have so permeated modern culture that it now is widely taken to characterize our times. We live in an 'information society', an 'age of information'. Indeed, we look to models of information processing to explain our own patterns of thought.

The computer has played the central role in that transformation, both accommodating and encouraging ever broader views of 'information' and of how it can be transformed and communicated over time and space. Since the 1950s the computer has replaced traditional methods of accounting and record-keeping by a new industry of data processing. As a primary vehicle of communication over both space and time, it has come to form the core of modern information technology. What the English-speaking world refers to as "computer science" is known to the rest of western Europe as *informatique* (or *Informatik* or *informatica*). Much of the concern over information as a commodity and as a natural resource derives from the computer and from computer-based communications technology.[1] Hence, the history of the computer and of computing is central to that of information science and technology, providing a thread by which to maintain bearing while exploring the ever-growing maze of disciplines and subdisciplines that claim information as their subject.

Despite the pervasive presence of computing in modern science and technology, not to mention modern society itself, the history of computing has yet to establish a significant presence in the history of science

[1] To characterize the unprecedented capabilities of computers linked to telecommunications, Nora and Minc (1978) coined the term *télématique*.

and technology. Meetings of the History of Science Society and the Society for the History of Technology in recent years have included very few sessions devoted specifically to history of computing, and few of the thematic sessions have included contributions from the perspective of computing. There is clearly a balance to be redressed here.

The status of the history of computing within the history of technology surely reflects on both parties, but the bulk of the task of redress lies with the former. A look at the literature shows that, by and large, historians of computing are addressing few of the questions that historians of technology are now asking. It is worth looking at what those questions are and what form they might take when addressed to computing. The question is how to bring the history of computing into line with what should be its parent discipline. Doing so will follow a two-way street: the history of computing should use models from the history of technology at the same time that we use the history of computing to test those models. In some aspects, at least, computing poses some of the major questions of the history of technology in special ways. Both fields have much to learn from the other.

Computing's Present History

Where the current literature in the history of computing is self-consciously historical, it focuses in large part on hardware and on the pre-history and early development of the computer.[2] Where it touches on later developments or provides a wider view, it is only incidentally historical. A major portion of the literature stems from the people involved, either through regular surveys of the state and development of various fields (e.g. Rosen 1967, Sammet 1969)[3] or compilations of seminal papers (Randell 1982; Yourdon 1979, 1982; AT&T 1987),[4] or through reminiscences and retrospectives, either written directly or transcribed from their contributions to conferences and symposia.[5] Biographies of men or machines --some heroic, some polemical, some both-- are a prominent genre, and one reads a lot about "pioneers". A few corporate histories have appeared, most notably *IBM's Early Computers* (Bashe et al. 1986), but they too are in-house productions.

This literature represents for the most part "insider" history, full of facts and firsts. While it is first-hand and expert, it is also guided by the current state of knowledge and bound by the professional culture. That is, its authors take as givens (often technical givens) what a more critical, outside viewer might see as choices. Reading their accounts makes it difficult to see the alternatives, as the authors themselves lose touch with a time when they did not know what they now know. In the long run, most of this literature will become primary sources, if not of the development of computing per se, then of its emerging culture.

From the outset, the computer attracted the attention of journalists, who by the late '50s were beginning to recount its history. The result is a sizable inventory of

[2] See Aspray (1984) for a recent, brief survey of the state of the field.

[3] Many of the articles in *Computing Surveys*, begun in 1969, include an historical review of the subject.

[4] The 25th-anniversary issues of the leading journals also contain useful collections of important articles.

[5] Wexelblatt (1981), a record of the 1978 ACM Conference on the History of Programming Languages, is an excellent example, as is a recent issue of the *Annals of the History of Computing* on the Burroughs B5000.

accounts having the virtues and vices of the journalist's craft. They are vivid, they capture the spirit of the people and of the institutions they portray, and they have an eye for the telling anecdote. But their immediacy comes at the price of perspective. Written by people more or less knowledgeable about the subject and about the history of technology, these accounts tend to focus on the unusual and the spectacular, be it people or lines of research, and they often cede to the self-evaluation of their subjects. Thus the microcomputer and artificial intelligence have had the lion's share of attention, as their advocates have roared a succession of millenia.

The journalistic accounts veer into another major portion of the literature on computing, namely what may be called "social impact statements". Often difficult to distinguish from futurist musing on the computer, the discussions of the effects of the computer on society and its various activities tend on the whole to view computing apart from the history of technology rather than from its perspective. History here serves the purpose of social analysis, criticism, and commentary. Hence much of it comes from popular accounts taken uncritically and episodically to support non-historical, often polemical, theses. Some of this literature rests on a frankly political agenda; whether its models and modes of analysis provide insight depends on whether one agrees with that agenda.

Finally, there is a small body of professionally historical work, dealing for the most part with the origins of the computer, its invention and early development (e.g. Stern 1981, Ceruzzi 1982, Williams 1986). It is meant as no denigration of that work to note that it stops at the point where computing becomes a significant presence in science, technology, and society. There historians stand before the daunting complexity of a subject that has grown exponentially in size and variety, looking not so much like an uncharted ocean as like a trackless jungle. We pace on the edge, pondering where to cut in.

The Questions of the History of Technology

The state of the literature in history of computing emerges perhaps more clearly by comparison (and by contrast) with what is currently appearing in the history of technology in general and with the questions that have occupied historians of technology over the past decade or so. Those questions derive from a cluster of seminal articles by George S. Daniels, Edwin T. Layton, Jr., Eugene S. Ferguson, Nathan Rosenberg, and Thomas P. Hughes, among others. How has the relationship between science and technology changed and developed over time and place? How has engineering evolved, both as an intellectual activity and as a social role? Is technology the creator of demand or a response to it? Put another way, does technology follow a society's momentum or redirect it by external impulse?[6] How far does economics go in explaining technological innovation and development? How do new technologies establish themselves in society, and how does society adapt to them? To what extent and in what ways do societies engender new technologies? What are the patterns by which technology is transferred from one culture to another? What role do governments play in fostering and directing technological

[6] George Daniels (1970) put the question as an assertion (p.6): "... the real effect of technical innovation [has been] to help Americans do better what they had already shown a marked inclination to do." The seeming "social lag" in adapting to new technology, he argued, is more likely economic in nature.

innovation and development? These are some of the "big questions", as George Daniels (1970) once put it. They can be broken down into smaller, more manageable questions, but ultimately they are the questions for which historians of technology bear special responsibility within the historical community. They are all of them questions which can shed light on the development of computing while it in turn elucidates them.

A few examples from recent literature must suffice to suggest the approaches historians of technology are taking to those questions. Each suggests by implication what might be done in the history of computing. A spate of studies on industrial research laboratories has explored the sources, purposes and strategies of organized innovation, invention, and patenting in the late 19th and early 20th centuries, bringing out the dynamics of technological improvement that Rosenberg (1979) suggested was a major source of growth in productivity. In *Networks of Power* Thomas P. Hughes (1983) has provided a model for pursuing another suggestion by Rosenberg, namely the need to treat technologies as interactive constituents of systems. Developments in one subsystem may be responses to demands in others and hence have their real pay-offs there. Or a breakthrough in one component of the system may unexpectedly create new opportunities in the others, or even force a reorganization of the system itself.

In detailed examinations of one of the "really big questions" of the history of American technology, Merritt Roe Smith (1977) and David A. Hounshell (1984) have traced the origins of the "American System" and its evolution into mass production and the assembly line. Both have entered the workshops and factories to reveal the quite uneven reception and progress of that system, never so monolithic or pervasive as it seemed then or has seemed since. Daniel Nelson (1975) and Stephen Meyer (1981) have entered the factory floor by another door to study the effects of mass production on the workers it organized.

Looking at technology in other contexts, Walter McDougall (1985) has anatomized the means and motivation of government support of research and development since World War II, revealing structures and patterns that extend well beyond the space program. Behind his study stands the ongoing history of NASA and of its individual projects. From another perspective, David F. Noble (1984) has examined the "command technology" that lay behind the development of numerically controlled tools. At a more mundane level, Ruth Cowan (1983) has shown how "progress is our most important product" often translated into *More Work for Mother*, while her own experiments in early nineteenth-century domestic technology have brought out the intimate relationship between household work and family relations.

In the late 1970s Anthony F.C. Wallace (1978) and Eugene Ferguson (1979b) recalled our attention to the non-verbal modes of thought that seem more characteristic of the inventor and engineer than does the language-based thinking of the scientist.[7] Brooke Hindle's (1981) study of Morse's telegraph and Reese Jenkins's (1987) recent work on the iconic patterns of Edison's thought provide examples of the insights historians can derive from artifacts read as the concrete expressions of visual and tactile cognition, recognizing that, as Henry Ford once put it,

[7] See in particular Wallace's "Thinking About Machinery" (Wallace 1978, pp.237ff.).

> There is an immense amount to be learned simply by tinkering with things. It is not possible to learn from books how everything is made --and a real mechanic ought to know how nearly everything is made. Machines are to a mechanic what books are to a writer. He gets ideas from them, and if he has any brains he will apply those ideas (Ford 1922, p.24).[8]

The renewed emphasis on the visual has reinforced the natural ties between the historian of technology and the museum, at the same time that it has forged links between history of technology and the study of material culture.

The Tripartite Nature of Computing

Before trying to translate some of the above questions and models into forms specific to the history of computing, it may help to reflect a bit on the complexity of the object of our study. The computer is not one thing, but many different things, and the same holds true of computing. There is about both terms a deceptive singularity to which we fall victim when, as is now common, we prematurely unite its multiple historical sources into a single stream, treating Charles Babbage's analytical engine and George Boole's algebra of thought as if they were conceptually related by something other than 20th-century hindsight. Whatever John von Neumann's precise role in designing the "von Neumann architecture" that defines the computer for the period with which historians are properly concerned, it is really only in von Neumann's collaboration with the ENIAC team that two quite separate historical strands came together: the effort to achieve high-speed, high-precision, automatic calculation and the effort to design a logic machine capable of significant reasoning.[9]

The dual nature of the computer is reflected in its dual origins: hardware in the sequence of devices that stretches from the Pascaline to the ENIAC, software in the series of investigations that reaches from Leibniz's combinatorics to Turing's abstract machines. Until the two strands come together in the computer, they belong to different histories, the electronic calculator to the history of technology, the logic machine to the history of mathematics,[10] and they can be unfolded separately without significant loss of fullness or texture. Though they come together in the computer, they do not unite. The computer remains an amalgam of technological device and mathematical concept, which retain separate identities despite their influence on one another.

Thus the computer in itself embodies one of the central problems of the history of technology, namely the relation of science and

[8] In *The Sciences of the Artificial* Herbert Simon (1981; cf. Newell and Simon 1976) argues forcefully for the empirical nature of computer research that underlies its mathematical trappings. The thinking of computer designers and programmers is embodied in the way their machines and programs *work*, and the languages they use to specify how things are to work are themselves artifacts. The models they use are filled with images difficult or distractingly tedious to translate into words; cf. Bolter (1984).

[9] I do not make this claim in ignorance of Konrad Zuse's Z4 or Alan Turing's ACE, which realized roughly the same goals as von Neumann's along independent paths. Clearly the computer was "in the air" by the 1940s. But it was the *19*40s, not the 1840s.

[10] I am including the history of mathematical logic in the history of mathematics

technology.[11] Computing as an enterprise deepens the problem. For not only are finite automata or denotational semantics independent of integrated circuits, they are also linked in only the most tenuous and uncertain way to programs and programming, that is, to software and its production. Since the mid-1960s experience in this realm has revealed a third strand in the nature of the computer. Between the mathematics that makes the device theoretically possible and the electronics that makes it practically feasible lies the programming that makes it intellectually, economically, and socially useful. Unlike the extremes, the middle remains a craft, technical rather than technological, mathematical only in appearance. It poses the question of the relation of science and technology in a very special form.

That tripartite structure shows up in the three distinct disciplines that are concerned with the computer: electrical engineering, computer science, and software engineering. Of these, the first is the most well established, since it predates the computer, even though its current focus on microelectronics reflects its basic orientation toward the device. Computer science began to take shape during the 1960s, as it brought together common concerns from mathematical logic (automata, proof theory, recursive function theory), mathematical linguistics, and numerical analysis (algorithms, computational complexity), adding to them questions of the organization of information (data structures) and the relation of computer architecture to patterns of computation. Software engineering, conceived as a deliberately provocative term in 1967 (Naur and Randell 1969), has developed more as a set of techniques than as a body of learning. Except for a few university centers, such as Carnegie-Mellon University, University of North Carolina, Berkeley, and Oxford, it remains primarily a concern of military and industrial R&D aimed at the design and implementation of large, complex systems, and the driving forces are cost and reliability.

History of Computing as History of Technology

Consider, then, the history of computing in light of current history of technology. Several lines of inquiry seem particularly promising. Studies such as those cited above offer a panoply of models for tracing the patterns of growth and progress in computing as a technology. It is worth asking, for example, whether the computing industry has moved forward more by big advances of radical innovation or by small steps of improvement. Has it followed the process described by Nathan Rosenberg, whereby "... technological improvement not only enters the structure of the economy through the main entrance, as when it takes the highly visible form of major patentable technological breakthroughs, but that it also employs numerous and less visible side and rear entrances where its arrival is unobtrusive, unannounced, unobserved, and uncelebrated" (Rosenberg 1979, p.26)? To determine whether that is the case will require changes in the history of computing as it is currently practiced. It will mean looking beyond "firsts" to the revisions and modifications that made products work and that account for their real impact. Given the corporate, collaborative structure of modern

[11] It should sharpen the question for the history of science as well, if only by giving special force to the reciprocal influence of scientific theory and scientific instrumentation. But up to now at least it has not attracted the same attention. The computer may well change that as the shaping of scientific concepts and the pursuit of scientific inquiry come to depend on the state of computer technology.

R&D, historians of computing must follow the admonition once made to historians of technology to stop "substituting biography for careful analysis of social processes". Without denigrating the role of heroes and pioneers, we need more knowledge of computing's equivalent of "shop practices, [and of] the activities of lower-level technicians in factories" (Daniels 1970, p.11). The question is how to pursue that inquiry across the variegated range of the emerging industry.

Viewing computing both as a system in itself and as a component of a variety of larger systems may provide important insights into the the dynamics of its development and may help to distinguish between its internal and its external history. For example, it suggests an approach to the question of the relation between hardware and software, often couched in the antagonistic form of one driving the other, a form which seems to assume that the two are relatively independent of one another. By contrast, linking them in a system emphasizes their mutual dependence. One expects of a system that the relationship among its internal components and their relationships to external components will vary over time and place but that they will do so in a way that maintains a certain equilibrium or homeostasis, even as the system itself evolves. Seen in that light, the relation between hardware and software is a question not so much of driving forces, or of stimulus and response, as of constraints and degrees of freedom. While in principle all computers have the same capacities as universal Turing machines, in practice different architectures are conducive to different forms of computing. Certain architectures have technical thresholds (e.g. VSLI is a prerequisite to massively parallel computing), others reflect conscious choices among equally feasible alternatives; some have been influenced by the needs and concerns of software production, others by the special purposes of customers. Early on, programming had to conform to the narrow limits of speed and memory set by vacuum-tube circuitry. As largely exogenous factors in the electronics industry made it possible to expand those limits, and at the same time drastically lowered the cost of hardware, programming could take practical advantage of research into programming languages and compilers. Researchers' ideas of multiuser systems, interactive programming, or virtual memory required advances in hardware at the same time that they drew out the full power of a new generation of machines. Just as new architectures have challenged established forms of programming, so too theoretical advances in computation and artificial intelligence have suggested new ways of organizing processors (e.g. Backus 1977).

At present, the evolution of computing as a system and of its interfaces with other systems of thought and action has yet to be traced. Indeed, it is not clear how many identifiable systems constitute computing itself, given the diverse contexts in which it has developed. We speak of the computer industry as if it were a monolith rather than a network of interdependent industries with separate interests and concerns. In addition to historically more analytical studies of individual firms, both large and small, we need analyses of their interaction and interdependence. The same holds for government and academia, neither of which has spoken with one voice on matters of computing. Of particular interest here may be the system-building role of the computer in forging new links of interdependence among universities, government, and industry after World War II.

Arguing in "The Big Questions" that creators of the machinery underpinning the American System worked from a knowledge of the entire sequence of operations in production,[12] Daniels (1970) pointed to Peter Drucker's suggestion that "the organization of work be used as a unifying concept in the history of technology." The recent volume by Charles Bashe et al. on *IBM's Early Computers* illustrates the potential fruitfulness of that suggestion for the history of computing. In tracing IBM's adaptation to the computer, they bring out the corporate tensions and adjustments introduced into IBM by the need to keep abreast of fast-breaking developments in science and technology and in turn to share its research with others.[13] The computer reshaped R&D at IBM, defining new relations between marketing and research, introducing a new breed of scientific personnel with new ways of doing things, and creating new roles, in particular that of the programmer. Whether the same holds true of, say, Bell Laboratories or G.E. Research Laboratories, remains to be studied, as does the structure of the R&D institutions established by the many new firms that constituted the growing computer industry of the '50s, '60s, and '70s. Tracy Kidder's (1981) frankly journalistic account of development at Data General has given us a tantalizing glimpse of the patterns we may find. Equally important will be studies of the emergence of the data-processing shop, whether as an independent computer service or as a new element in established institutions.[14] More than one company found that the computer reorganized *de facto* the lines of effective managerial power.

The computer seems an obvious place to look for insight into the question of whether new technologies respond to need or create it. Clearly, the first computers responded to the felt need for high-speed, automatic calculation, and that remained the justification for their early development during the late '40s. Indeed, the numerical analysts evidently considered the computer to be their baby and resented its adoption by "computerologists" in the late '50s and early '60s (Wilkinson 1971). But it seems equally clear that the computer became the core of an emergent data-processing industry more by creating demand than by responding to it. Much as Henry Ford taught the nation how to use an automobile, IBM and its competitors taught the nation's businesses (and its government) how to use the computer. How much of the technical development of the computer originated in the marketing division remains an untold story central to an understanding of modern technology.[15] Kidder's *Soul of a New Machine* again offers a glimpse of what that story may reveal.

One major factor in the creation of demand seems to have been the alliance

[12] Elting E. Morison (1974) has pursued this point along slightly different but equally revealing lines.

[13] Lundstrom (1987) has recently chronicled the failure of some companies to make the requisite adjustments.

[14] The obvious citations here are Kraft (1977) and Greenbaum (1979), but both works are concerned more with politics than with computing, and the focus of their political concerns, the "deskilling" of programmers through the imposition of methods of structured programming, has proved ephemeral, as subsequent experience and data show that programmers have made the transition with no significant loss of control over their work; cf. Boehm (1981).

[15] See, for example, Burke (1970): "Thus technological innovation is not the product of society as a whole but emanates rather from certain segments within or outside of it; the men or institutions responsible for the innovation, to be successful, must 'sell' it to the general public; and innovation *does* have the effect of creating broad social change.(p.23)" Ferguson (1979a) has made a similar observation about selling new technology.

between the computer and the nascent field of operations research/management science. As the pages of the *Harvard Business Review* for 1953 show, the computer and operations research hit the business stage together, each a new and untried tool of management, both clothed in the mantle of science. Against the fanciful backdrop of Croesus' defeat by camel-riding Persians, an IBM advertisement proclaimed that "Yesterday ... 'The Fates' Decided. Today ... Facts Are What Count". Appealing to fact-based strides in "military science, pure science, commerce, and industry", the advertisement pointed beyond data processing to "'mathematical models' of specific processes, products, or situations, [by which] man today can predetermine probable results, minimize risks and costs." In less vivid terms, Cyril C. Herrmann of MIT and John F. Magee of Arthur D. Little introduced readers of *HBR* to "'Operations Research' for Management" (1953), and John Diebold (1953) proclaimed "Automation - The New Technology". As Herbert Simon (1960, p.14) later pointed out, operations research was both old and new, with roots going back to Charles Babbage and Frederick W. Taylor. Its novelty lay precisely in its claim to provide 'mathematical models' of business operations as a basis for rational decision-making. Depending for their sensitivity on computationally intensive algorithms and large volumes of data, those models required the power of the computer.

It seems crucial for the development of the computer industry that the business community accepted the joint claims of OR and the computer long before either could validate them by, say, cost-benefit analysis. The decision to adopt the new methods of "rational decision-making" seems itself to have been less than fully rational:

> As business managers we are revolutionizing the procedures of our factories and offices with automation, but what about out decision making? In other words, isn't there a danger that our thought processes will be left in the horse-and-buggy stage while our operations are being run in the age of nucleonics, electronics, and jet propulsion? ... Are the engineering and scientific symbols of our age significant indicators of a need for change? (Hurni 1955, p.49)

Even at this early stage, the computer had acquired symbolic force in the business community and in society at large. We need to know the sources of that force and how it worked to weave the computer into the economic and social fabric.[16]

The government has played a determining role in at least four areas of computing: microelectronics; interactive, real-time systems; artificial intelligence; and software engineering. None of these stories has been told by an historian, although each promises deep insight into the issues raised above. Modern weapons systems and the space program placed a premium on miniaturization of circuits. Given the costs of research, development, and tooling for production, it is hard to imagine that the

[16] Along these lines, historians of computing would do well to remember that a line of writings on the nature, impact, and even history of computing stretching from Edmund C. Berkeley's (1949) *Giant Brains* through John Diebold's several volumes to Edward Feigenbaum's and Pamela McCorduck's (1983) The Fifth Generation stems from people with a product to sell, whether management consulting or expert systems.

integrated circuit and the microprocessor would have emerged --at least as quickly as they did-- without government support. As Frank Rose (1984) put it in *Into the Heart of the Mind*, "The computerization of society ... has essentially been a side effect of the computerization of war.(p.36)" More is involved than smaller computers. Architecture and software change in response to speed of processor and size of memory. As a result, the rapid pace of miniaturization tended to place already inadequate methods of software production under the pressure of rising expectations. By the early 1970s the Department of Defense, as the nation's single largest procurer of software, had declared a major stake in the development of software engineering as a body of methods and tools for reducing the costs and increasing the reliability of large programs.

As Howard Rheingold (1985) has described in *Tools for Thought* the government was quick to seize on the interest of computer scientists at MIT in developing the computer as an enhancement and extension of human intellectual capabilities. In general, that interest coincided with the needs of national defense in the form of interactive computing, visual displays of both text and graphics, multi-user systems, and inter-computer networks. The Advanced Research Projects Agency (later DARPA), soon became a source of almost unlimited funding for research in these areas, a source that bypassed the usual procedures of scientific funding, in particular peer review. Much of the early research in artificial intelligence derived its funding from the same source, and its development as a field of computer science surely reflects that independence from the agenda of the discipline as a whole.

Although we commonly speak of hardware and software in tandem, it is worth noting that in a strict sense the notion of software is an artifact of computing in the business and government sectors during the '50s. Only when the computer left the research laboratory and the hands of the scientists and engineers did the writing of programs become a question of production. It is in that light that we may most fruitfully view the development of programming languages, programming systems, operating systems, database and file management systems, and communications and networks, all of them aimed at facilitating the work of programmers, maintaining managerial control over them, and assuring the reliability of their programs. The Babel of programming languages in the '60s tends to distract attention from the fact that three of the most commonly used languages today are also among the oldest: FORTRAN for scientific computing, COBOL for data processing, and LISP for artificial intelligence. ALGOL might have remained a laboratory language had it and its offspring not become the vehicles of structured programming, a movement addressed directly to the problems of programming as a form of production.[17]

Central to the history of software is the sense of "crisis" that emerged in the late '60s as one large project after another ran behind schedule, over budget, and below specifications. Though pervasive throughout the industry, it posed enough of a strategic threat for the NATO Science Committee to convene an international conference in 1968

[17] An effort at international cooperation in establishing a standard programming language, ALGOL from its inception in 1956 to its final (and, some argued, over-refined) form in 1968 provides a multileveled view of computing in the '60s. While contributing richly to the conceptual development of programming languages, it also has a political history which carries down to the present in differing directions of research, both in computer science and, perhaps most clearly, in software engineering.

to address it. To emphasize the need for a concerted effort along new lines, the committee coined the term "software engineering", reflecting the view that the problem required the combination of science and management thought characteristic of engineering. Efforts to define that combination and to develop the corresponding methods constitute much of the history of computing during the 1970s, at least in the realm of large systems, and it is the essential background to the story of Ada in the 1980s. It also reveals apparently fundamental differences between the formal, mathematical orientation of European computer scientists and the practical, industrial focus of their American counterparts. Historians of science and technology have seen those differences in the past and have sought to explain them. Can historians of computing use those explanations and in turn help to articulate them?

The effort to give meaning to "software engineering" as a discipline and to define a place for it in the training of computer professionals should call the historian's attention to the constellation of questions contained under the heading of "discipline formation and professionalization". In 1950 computing consisted of a handful of specially designed machines and a handful of specially trained programmers. By 1955 some 1000 general-purpose computers required the services of some 10,000 programmers. By 1960, the number of devices had increased fivefold, the number of programmers sixfold. And so the growth continued. With it came associations, societies, journals, magazines, and claims to professional and academic standing. The development of these institutions is an essential part of the the social history of computing as a technological enterprise. Again, one may ask to what extent that development has followed historical patterns of institutionalization and to what extent it has created its own.

The question of sources illustrates particularly well how recent work in the history of technology may provide important guidance to the history of computing, at the same time that the latter adds new perspectives to that work. As noted above, historians of technology have focused new attention on the non-verbal expressions of engineering practice. Of the three main strands of computing, only theoretical computer science is essentially verbal in nature. Its sources come in the form most familiar to historians of science, namely books, articles, and other less formal pieces of writing, which by and large encompass the thinking behind them. We know pretty well how to read them, even for what they do not say explicitly. Similarly, at the level of institutional and social history, we seem to be on familiar ground, suffering largely from an embarrassment of wealth unwinnowed by time.

But the computers themselves and the programs that were written for them constitute a quite different range of sources and thus pose the challenge of determining how to read them. As artifacts, computers present the problem of all electrical and electronic devices. They are machines without moving parts. Even when they are running, they display no internal action to explain their outward behavior. Yet, Tracy Kidder's (1981) portrait of Tom West sneaking a look at the boards of the new Vax to see how DEC had gone about its work reminds us that the actual machines may hold tales untold by manuals, technical reports, and engineering drawings. Those sources too demand our attention. When imaginatively read, they promise to throw light not only on the designers but also on those for whom they were designing. Through the hardware and its attendant sources one can follow the changing physiognomy of computers as they made their way from the laboratories and large installations to the office and the home.

Today's prototypical computer iconically links television to typewriter. How that form emerged from a roomful of tubes and switches is a matter of both technical and cultural history.

Though hard to interpret, the hardware is at least tangible. Software by contrast is elusively intangible. In essence, it is the behavior of the machines when running. It is what converts their architecture to action, and it is constructed with action in mind; the programmer aims to make something happen. What, then, captures software for the historical record? How do we document and preserve an historically significant compiler, operating sysem, or database? Computer scientists have pointed to the limitations of the static program text as a basis for determining the program's dynamic behavior, and a provocative article (DeMillo et al. 1979) has questioned how much the written record of programming can tell us about the behavior of programmers. Yet, Gerald M. Weinberg (1971, Chapter 1) has given an example of how programs may be read to reveal the machines and people behind them. In a sense, historians of computing encounter from the opposite direction the problem faced by the software industry: what constitutes an adequate and reliable surrogate for an actually running program? How, in particular, does the historian recapture, or the producer anticipate, the component that is always missing from the static record of software, namely the user for whom it is written and whose behavior is an essential part of it?

Placing the history of computing in the context of the history of technology promises a peculiarly recursive benefit. Although computation by machines has a long history, computing in the sense I have been using here did not exist before the late 1940s. There were no computers, no programmers, no computer scientists, no computer managers. Hence those who invented and improved the computer, those who determined how to program it, those who defined its scientific foundations, those who established it as an industry in itself and introduced it into business and industry all came to computing from some other background. With no inherent precedents for their work, they had to find their own precedents. Much of the history of computing, certainly for the first generation, but probably also for the second and third, derives from the precedents these people drew from their past experience. In that sense, the history of technology shaped the history of computing, and the history of computing must turn to the history of technology for initial bearings.

A specific example may help to illustrate the point. Daniels (1970) stated as one of the really big questions the development of the 'American System' and its culmination in mass production. It is perhaps the central fact of technology in 19th-century America, and every historian of the subject must grapple with it. So too, though Daniels did not make the point, must historians of 20th-century technology. For mass production has become an historical touchstone for modern engineers, in the area of software as well as elsewhere.

For instance, in one of the major invited papers at the NATO Software Engineering Conference of 1968, M.D. McIlroy of Bell Telephone Laboratories looked forward to the end of a "preindustrial era" in programming. His metaphors and similes harked back to the machine-tool industry and its methods of production.

> We undoubtedly produce software by backward techniques. We undoubtedly get the short end of the stick in confrontations with hardware people because they are the

> industrialists and we are the crofters. Software production today appears in the scale of industrialization somewhere below the more backward construction industries. I think its proper place is considerably higher, and would like to investigate the prospects for mass-production techniques in software.(McIlroy, 1969)

What McIlroy had in mind was not replication in large numbers, which is trivial for the computer, but rather programmed modules that might serve as standardized, interchangeable parts to be drawn from the library shelf and inserted in larger production programs. A quotation from McIlroy's paper served as *leitmotiv* to the first part of Peter Wegner's series on "Capital Intensive Software Technology" in the July 1984 number of *IEEE Software*, which was richly illustrated by photographs of capital industry in the 1930s and included insets on the history of technology.[18] By then McIlroy's equivalent to interchangeable parts had become "reusable software" and software engineers had developed more sophisticated tools for producing it. Whether they were (or now are) any closer to the goal is less important to the historian than the continuing strength of the model. It reveals historical self-consciousness.

We should appreciate that self-consciousness at the same time that we view it critically, resisting the temptation to accept the comparisons as valid. An activity's choice of historical models is itself part of the history of the activity. McIlroy was not describing the state or even the direction of software in 1968. Rather, he was proposing an historical precedent on which to base its future development. What is of interest to the historian of computing is why McIlroy chose the model of mass production as that precedent. Precisely what model of mass production did he have in mind, why did he think it appropriate or applicable to software, why did he think his audience would respond well to the proposal, and so on? The history of technology provides a critical context for evaluating the answers, indeed for shaping the questions. For historians, too, the evolving techniques of mass production in the 19th century constitute a model, or prototype, of technological development. Whether it is one model or a set of closely related models is a matter of current scholarly debate, but some features seem clear. As a system it rested on foundations established in the early and mid-19th century, among them in particular the development of the machine-tool industry, which, as Nathan Rosenberg (1963) has shown, itself followed a characteristic and revealing pattern of innovation and diffusion of new techniques. Even with the requisite precision machinery, methods of mass production did not transfer directly or easily from one industry to another, and its introduction often took place in stages peculiar to production process involved (Hounshell 1984). Software production may prove to be the latest variation of the model, or critical history of technology may show how it has not fit.

Conclusion: The Real Computer Revolution

We can take this example a step farther. From various perspectives, people have been drawn to compare the computer to the automobile.

[18] One has to wonder about an article on software engineering that envisions progress on an industrial model and uses photographs taken from the Great Depression.

Apple, Atari, and others have boasted of creating the Model T of microcomputers, clearly intending to convey the image of a car in every garage, an automobile that everyone could drive, a machine that reshaped American life. The software engineers who invoke the image of mass production have it inseparably linked in their minds to the automobile and its interchangeable variations on a standard theme.

The two analogies serve different aims within the computer industry, the first looking to the microcomputer as an object of mass consumption, the second to software systems as objects of mass production. But they share the vision of a society radically altered by a new technology. Beneath the comparison lies the conviction that the computer is bringing about a revolution as profound as that triggered by the automobile. The comparison between the machines is fascinating in itself. Just how does one weigh the PC against the PT (personal transporter)?[19] For that matter, which PC is the Model T: the Apple][, the IBM, the Atari ST, the Macintosh? Yet the question is deeper than that. What would it mean for a microcomputer to play the role of the Model T in determining new social, economic, and political patterns? The historical term in that comparison is not the Model T, but Middletown (Lynd and Lynd 1929), where in less than forty years "high-speed steel and Ford cars" had fundamentally changed the nature of work and the lives of the workers. Where is the Middletown of today, similarly transformed by the presence of the microcomputer? Where would one look? How would one identify the changes? What patterns of social and intellectual behavior mark such transformation? In short, how does one compare technological societies? That is one of the "big questions" for historians of technology, and it is only in the context of the history of technology that it will be answered for the computer.

From the very beginning, the computer has borne the label "revolutionary". Even as the first commercial machines were being delivered, commentators were extolling or fretting over the radical changes the widespread use of computers would entail, and few doubted their use would be widespread. The computer directed people's eyes toward the future, and a few thousand bytes of memory seemed space enough for the solution of almost any problem. On that both enthusiasts and critics could agree. Computing meant unprecedented power for science, industry, and business, and with the power came difficulties and dangers that seemed equally unprecedented. By its nature as well as by its youth, the computer appeared to have no history.

Yet, "revolution" is an essentially historical concept (Cohen 1986). Even when turning things on their head, one can only define what is new by what is old, and innovation, however imaginative, can only proceed from what exists. The computer had a history out of which it emerged as a new device, and computing took shape from other, continuing activities, each with its own historical momentum. As the world of the computer acquired its own form, it remained embedded in the worlds of science, technology, industry, and business which structured computing even as they changed in response to it. In doing so they linked the history of computing to their own histories, which in turn reflected the presence of a fundamentally new resource.

What is truly revolutionary about the computer will become clear only when computing acquires a proper history, one that

[19] The latter designation stems from Frand (1983).

ties it to other technologies and thus uncovers the precedents that make its innovations significant. Pursued within the larger enterprise of the history of technology, the history of computing will acquire the context of place and time that gives history meaning.

Acknowledgements

This article is a slightly revised version of a position paper prepared for the Seminar on Information Technologies in Historical Context, held at the National Museum of American History, 11 September 1987. It benefitted at that time from the critical comments of David K. Allison, William Aspray, I. Bernard Cohen, and Arthur Norberg. The research from which it stems has been generously supported by the Alfred P. Sloan, Jr. Foundation under its New Liberal Arts Program.

References

Aspray, William. 1984. "Literature and Institutions in the History of Computing." *ISIS* 75, pp. 162-170.

AT&T Bell Laboratories. 1987. *UNIX System Readings and Applications.* 2 vols. Englewood Cliffs, N.J., Prentice-Hall.

Backus, John. 1977. "Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs." (ACM Turing Award Lecture for 1977). *Communications of the ACM*, 21,8, pp. 613-641.

Bashe, Charles J., Lyle R. Johnson, John H. Palmer, and Emerson W. Pugh. 1986. *IBM's Early Computers.* Cambridge, MA, MIT Press.

Berkeley, Edmund C. 1949. *Giant Brains or Machines That Think.* New York, John Wiley & Sons.

Bolter, J. David. 1984. *Turing's Man.* Chapel Hill, University of North Carolina Press.

Boehm, Barry. 1981. *Software Engineering Economics.* Englewood Cliffs, N.J., Prentice-Hall.

Burke, John G. 1970. "Comment: The complex nature of explanation in the historiography of technology." *Technology and Culture*, 11, pp. 22-26.

Buxton, J.N. and Brian Randell (eds.). 1970 *Software Engineering Techniques: Report on a conference sponsored by the NATO Science Committee, Rome, Italy, 27th to 31st October 1969.* Brussels, Scientific Affairs Department, NATO.
Cf. Naur et al. (1976).

Ceruzzi, Paul E. 1982 *Reckoners: The Prehistory of the Digital Computer, From Relays to the Stored Program Concept, 1935-1945.* Westport, CT, Greenwood Press.

Cohen, I. Bernard. 1986. *Revolutions in Science.* Cambridge, MA, Harvard University Press.

Cowan, Ruth. 1983. *More Work for Mother: The Ironies of Household Technology from the Open Hearth to the Microwave.* New York, Basic Books.

Daniels, George. 1970. "The Big Questions in the History of American Technology." *Technology and Culture*, 11, pp. 1-21.

DeMillo, Richard, Richard J. Lipton, and Alan J. Perlis. 1979. "Social Processes and Proofs of Theorems and Programs." *Communications of the ACM*, 22, 5, pp. 271-280.

Diebold, John. November-December 1953. "Automation - the new technology. *Harvard Business Review.* pp. 63-71.

Feigenbaum, Edward and Pamela McCorduck. 1983 *The Fifth Generation: Artificial Intelligence and Japan's Computer Challenge to the World.* Reading, MA, Addison-Wesley.

Ferguson, Eugene S. 1979a. "The American-ness of American Technology." *Technology and Culture*, 20, .pp. 3-24.

Ferguson, Eugene S. 1979b. "The Mind's Eye: Nonverbal Thought in Technology." *Science*, 197, pp. 827-836.

Ford, Henry. 1922. *My Life and Times.* Garden City, N.Y., Doubleday.

Frand, Erwin. July 1983. "Thoughts on Product Development: Remembrance of Things Past." *Industrial Research and Development.* p. 23.

Greenbaum, Joan. 1979. *In the Name of Efficiency: Management Theory and Shopfloor Practice in Data Processing Work.* Philadelphia, Temple University Press.

Herrmann, Cyril C. and John F. Magee. July-August 1953. "'Operations Research' for Management." *Harvard Business Review.* pp. 100-112.

Hindle, Brooke. 1981. *Emulation and Invention.* New York, Basic Books.

Hounshell, David A. 1984. *From American System to Mass Production: The Development of Manufacturing Technology in the United States.* Baltimore, Johns Hopkins University Press.

Hughes, Thomas P. 1983. *Networks of Power: Electrification in Western Society, 1880-1930.* Baltimore, Johns Hopkins University Press.

Hurni, Melvin L. September-October 1955. "Decision Making in the Age of Automation." *Harvard Business Review* 34, pp. 49-58.

Jenkins, Reese V. 1987. "Words, Images, Artifacts and Sound: Documents for the History of Technology." *British Journal for the History of Science* 20, pp. 39-56.

Kidder, Tracy. 1981. *The Soul of a New Machine.* New York, Little, Brown & Co.

Kraft, Philip. 1977. *Programmers and Managers: The Routinization of Computer Programming in the United States.* New York, Springer Verlag.

Lundstrom, David E. 1987. *A Few Good Men From Univac.* Cambridge, MA, MIT Press.

Lynd, Robert S. and Helen Merrell Lynd. 1929. *Middletown.* New York, Harcourt, Brace & World.

Machlup, Fritz and Una Mansfeld. 1983. *The Study of Information: Interdisciplinary Messages.* New York, John Wiley & Sons.

McDougall, Walter A. 1985. *... The Heavens and the Earth: A Political History of the Space Age.* New York, Basic Books.

McIlroy, M. Douglas. 1969. "Mass-Produced Software." In Naur and Randell 1969.

Meyer, Stephen III. 1981. *The Five-Dollar Day.* Albany, State University of New York Press.

Morison, Elting E. 1974. *From Know-How to Nowhere: The Development of American Technology.* New York, Basic Books.

Naur, Peter and Brian Randell (eds.). 1969. *Software Engineering: Report on a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7th to 11th October, 1968.* Brussels, Scientific Affairs Division, NATO. See Naur et al. 1976.

Naur, Peter, Brian Randell and J.N. Buxton (eds.). 1976. *Software Engineering: Concepts and Techniques.* New York, Petrocelli/Charter.

Nelson, Daniel. 1975. *Managers and Workers: Origins of the New Factory System in the United States, 1880-1920.* Madison, University of Wisconsin Press.

Newell, Allen and Simon, Herbert A. 1976. "Computer Science as Empirical Inquiry." *Communications of the ACM* 19, pp.113-126.

Noble, David. 1984. *Forces of Production: A Social History of Industrial Automation.* New York, Alfred Knopf.

Nora, Simon and Alain Minc. 1978. *L'Informatisation de la société.* Paris, La documentation française. Translated into English under the title *The Computerization of Society.* Cambridge, MA, MIT Press, 1980.

Randell, Brian (ed.). 1982. *Origins of Digital Computers: Selected Papers.* Berlin/Heidelberg/New York, Springer Verlag. 3rd ed.

Rheingold, Howard. 1985. *Tools for Thought: The People and Ideas Behind the Next Computer Revolution.* New York, Simon and Schuster.

Rose, Frank. 1984. *Into the Heart of the Mind: An American Quest for Artificial Intelligence.* New York, Random House.

Rosen, Saul. 1967. "Programming Systems and Languages - A Historical Survey." In his *Programming Systems and Languages.* New York, McGraw-Hill.

Rosenberg, Nathan. 1963. "Technological Change in the Machine Tool Industry, 1840-1910." *Journal of Economic History* 23, pp. 414-443.

Rosenberg, Nathan. 1979. "Technological interdependence in the American Economy." *Technology and Culture* 20, pp. 25-50.

Sammet, Jean. 1969. *Programming Languages. History and Fundamentals.* Englewood Cliffs, N.J., Prentice-Hall.

Simon, Herbert A. 1960. *The New Science of Management Decision.* New York, Harper & Row.

Simon, Herbert. 1981. *The Sciences of the Artificial.* Cambridge, Mass., MIT Press. 2nd ed.

Smith, Merritt Roe. 1977. *Harper's Ferry Armory and the New Technology.* Ithaca, Cornell University Press.

Stern, Nancy. 1981. *From ENIAC to UNIVAC: An Appraisal of the Eckert-Mauchly Computers.* Billerica, MA, Digital Press.

Wallace, Anthony F.P. 1978. *Rockdale: The Growth of an American Village in the Early Industrial Revolution.* New York, Alfred A. Knopf. (Paperback ed. New York, W.W. Norton, 1980).

Weinberg, Gerald M. 1971. *The Psychology of Computer Programming.* New York, Van Nostrand Reinhold.

Wexelblatt, Richard L. 1981. *History of Programming Languages.* New York, Academic Press.

Wilkinson, James H. 1971. "Some Comments from a Numerical Analyst." (1970 Turing Award Lecture). *Journal of the ACM* 18,2(1971), 137-147;

Williams, Michael. 1986. *A History of Computing Technology.* Englewood Cliffs, N.J., Prentice-Hall.

Yourdon, Edward. 1979. *Classics of Software Engineering.* New York, Yourdon Press.

Yourdon, Edward. 1982. *Papers of the Revolution.* New York, Yourdon Press.