SESSION 1

PRESIDENTIAL ADDRESS

The dimensions of computing

JONATHAN VAUGHAN
Hamilton College, Clinton, New York

Computers have become ubiquitous in the psychology laboratory over the last two decades. This paper is an examination of the laboratory computer's evolution; general evolutionary principles are used to predict future changes.

My goal in this paper is to examine computing as it is used in psychological research, from the perspectives of the temporal, spatial, economic, and social dimensions. I will look at where we have come in the 20-odd years since the computer first made its presence felt in the discipline of psychology, as well as speculate about what we might expect in the years to come. My approach will be theoretical—that is, there will be little practical information in this paper. That is why it is called a presidential address.

The Temporal Dimension of Computing

First, some background: I began my exploration of the temporal dimension of computing in psychology by looking back about 10 years in the annals of the Society for Computers in Psychology (SCiP), which are recorded in the journal *Behavior Research Methods, Instruments, & Computers* (BRMIC). I soon homed in on N. John Castellan's 1980 presidential address (Castellan, 1981). In that paper, Castellan started out by looking back at the papers that had been presented over the preceding decade at the National Conference on the Use of On-Line Computers in Psychology (NCUOCP), of which SCiP is the successor. Castellan's seemed a good example to follow, so I looked back through the papers of the past 10 years, and arrived again at Castellan's 1980 presidential address. I read it once more. Following Castellan's good example again, I found myself in the situation of the systems analyst who could not get out of the shower. He had read and followed the instructions on a bottle of shampoo: "Lather. Rinse. Repeat." So it was not surprising that as I printed out a draft of my talk my ImageWriter jammed, the paper

being crumpled into the form of a Möbius strip. Obviously I was getting nowhere. So I abandoned this decidedly one-sided approach, and decided to talk about other dimensions of computing before returning to the temporal dimension.

The Spatial Dimension of Computing

Like the Möbius strip, the evolution of the computer has been governed by a cyclic design process. The principle behind this process, which has been called the Wheel of Reincarnation (Myer & Sutherland, 1968, cited in Bell, Mudge, & MacNamara, 1978) describes an evolutionary process that we have all observed in the computers that have come and gone in our laboratories and on our desktops. The example in Figure 1 is directed at the evolution of visual display processors, but the underlying principle is applicable to every kind of computing machinery, on all scales of measurement.

The principle has three parts: (1) As rapidly as evolving technology permits, routine processes will be off-loaded from the general-purpose processors of a main computer, by being delegated to increasingly intelligent peripherals, in order to marginally improve the performance of the main processor. (2) As the peripherals take on these additional responsibilities, they, in turn, must become more and more complex and sophisticated. (3) Eventually, it becomes economically justifiable to endow the peripheral itself with all the power of a general-purpose processor: an independent read/write memory, all the appropriate computational hardware, and even, sometimes, the same instruction set as that of the main computer. Once this three-part cycle has been completed, it begins again, this time at the level of the peripheral device. Note, in Figure 1, the successive increases in the capabilities of the "peripheral" as we move around the wheel in a clockwise direction. The driving force that rotates the Wheel of Reincarnation is the seesaw oscillation between which of the following is more economical: special-purpose hardware of increasing complexity, or a

Correspondence should be sent to the author at the Department of Psychology, Hamilton College, 198 College Hill Road, Clinton, NY 13323 (e-mail: jvaughan@hamilton.edu).

Figure 1. The Wheel of Reincarnation describes the evolution of increasingly sophisticated peripheral devices.

new dedicated general-purpose processor dedicated to the same functions.

The Wheel of Reincarnation has affected computing in psychology in a number of important ways. What was once a mechanical teletypewriter in a corner of the laboratory or a timesharing terminal on the desk has acquired more and more independent computing power, which affords more and more complicated operations, to lighten the load on the main processor. In psychology laboratories, first came the dumb and then the smart CRT terminals, followed by the desktop computers and the workstations. Each of these devices has grown its own intelligent peripherals. At the most general level, the outstanding example of the Wheel of Reincarnation has been the off-loading of computational cycles from the mainframe to the desktop. When we now log onto the campus electronic mail system with a terminal emulator in a multiprocessing operating system like the Macintosh Multifinder, we see both the vestiges of that early terminal and the impressive extra capacity that the erstwhile "terminal" has acquired.

Consider the progression from the once monolithic central campus computer center (Figure 2, top panel) to our present situation, in which our desktop machines are connected via local area networks to departmental servers, which are often further connected via wide area networks to campus-wide or intracampus networks (Figure 2, bottom panel). Viewed abstractly, the arborization of the Wheel of Reincarnation forms the kind of fractal pattern often observed in natural growth processes.

The Wheel of Reincarnation continues to turn, of course. As the various product lines have evolved, so has the functionality of their peripherals. In the original Macintosh, display memory was allocated out of the main machine memory, and all arithmetic computations were performed

by the main microprocessor. In later Macintosh computers, the display controller (on its own NuBus interface card) acquired its own memory, and floating computations were off-loaded to an arithmetic coprocessor. Now, the Macintosh mouse and keyboard both have embedded microprocessors (rather than only special-purpose circuitry), and many interface cards that plug into the Macintosh's NuBus expansion slots, or I/O boxes that connect via the serial port, themselves comprise complete general-purpose microprocessor systems. Similarly, printers are no longer passive peripherals that merely receive strings of characters or pixels. Most LaserWriters contain embedded 68000 processors, the same processor that is at the core of the Macintosh, and printing instructions are sent in the PostScript language, a programming language for the production of graphics.

These examples highlight some consequences of the Wheel of Reincarnation that may impinge on interfacing the computer in the laboratory. First, if the processor we are working with happens to fall in a certain phase of the Wheel of Reincarnation's rotation, then (ironically, in spite of all the computational power of that processor) the laboratory experimenter may have a more, not less, challenging job in communicating with the real world of sensors, response switches, and the external parapherna-

Mainframe

Mainframe with Timesharing Terminals

Mainframe, Minis with Peripherals

Network, Central Minis and Desktop Micros

Figure 2. Fractal growth of computing systems resulting from the successive rotations of the Wheel of Reincarnation.

Figure 3. The hypothetical allocation of campus space allocated to computing that results from successive rotations of the Wheel of Reincarnation. In each square, the space allocated to computing resources is given in black. Top panel: campus mainframe. Second panel: mainframe with timesharing terminals (one rotation of the Wheel of Reincarnation). Third panel: space allocated to computing resources after four rotations of the Wheel of Reincarnation. Bottom panel: eventual allocation of all space to computing.

lia of the laboratory than he or she did before the peripherals acquired their intelligence. The experimenter's program must take into account not only the experimental paradigm and the main computer's capabilities but also the syntax of communication with the intelligent peripherals. Furthermore, a sophisticated peripheral's operations may be asynchronous with the main program, and unpredictable interruptions of the main program may be required to service the increasingly independent intelligent peripherals.

A second consequence of the Wheel of Reincarnation has at least potentially ominous long-term consequences for our utilization of space. Let's start with a main computer, such as the mainframe in an old central campus computing center, and assume that the computer center occupies some arbitrary fraction (1/9, say) of usable campus research space. If the Wheel of Reincarnation

turns once, to delegate computation to the desktop, each of those desktop machines will require some fraction of the remaining laboratory or office space. In this example, each cycle of the Wheel of Reincarnation consumes another 1/9 of the remaining free space (Figure 3). Taking this evolution to its logical limit, other things begin equal, *all* space is at risk of being consumed by computing equipment and its peripherals.

We have avoided this grim outcome because other things are *not* equal. First, both the existing computing functions and the functions newly delegated to peripherals are likely to exploit ever smaller hardware, at least until miniaturization is constrained by the physical limits of the speed of light, the density of interconnections, and the ability to dissipate heat. Second, in order to reach the saturation limit in finite time, each cycle of the Wheel of Reincarnation would have to be achieved in ever shorter time, which would violate two equally rigid imperatives: No research grant gestates in fewer than 9 months, and no purchase order may clear the business office in less than 10 days.

Another constraint on the limitless expansion of computing resources is the countervailing inertia resulting from the need to exploit existing software investment (either a specific programming language or a computational instruction set). The need to amortize software investment was first formally expressed by Maurice Wilkes: "When a machine was finished, and a number of subroutines were in use, the order code could not be altered without causing a good deal of trouble. There would be almost as much capital sunk in the library of subroutines as [in] the machine itself, and builders of new machines in the future might wish to make use of the same order code as [that in] an existing machine in order that the subroutines could be taken over without modification" (cited in Bell et al., 1978, p. 26). Wilkes's insight was more prescient in 1949, before any digital computer was operating in this country, than it might seem now, but it is no less valid.

The Macintosh instruction set has been in production for about 7 years now, and it seems secure in the middle

PDP-5, 8, 12, Intercept Jr.

Instruction Set (1964-1976)

AND

TAD

DCA

JMS

JMP

IOT

IOP

Figure 4. The PDP-8 machine-language instruction set—an early "RISC" instruction set, although it was not thus labeled at the time (1964).

of its life cycle. To get an ideal of its expected useful lifetime, let us consider the Digital Equipment Corporation PDP series computers (the "other" PDP) that used an instruction set whose life cycle is clearly complete. This instruction set (Figure 4) was reimplemented 10 times with different technologies over a 15-year period. The PDP-5 was prototyped in 1964: it had the size of two refrigerators; the PDP-8 (1966), that of one small refrigerator. The 8/e (1970) had the size of a large breadbox. The single-chip LSI implementations of the instruction set were smaller still. The VT-78 processor fit in one corner of a CRT terminal housing, and the Intercept Jr. prototyping board was the size of a writing tablet (it used just four D-cell batteries for a power supply). All implementations ran essentially the same instruction set, shown as conventional three-letter acronyms (TLAs) in Figure 4. Fortunately, the PDP-8 operating systems that supported this invariant instruction set evolved tremendously over the intervening years, progressing through punched tape and magnetic tape to floppy and hard disk versions.

For current instruction sets such as that for the 680x0 processor of the Macintosh, now in its fourth implementation as the 68040, we may be near the limits of size and approaching the limits of speed improvement that are possible. Nevertheless, the cumulative investment in software might motivate the development of an economical, slower implementation of the instruction set (as occurred in the case of the Intersil implementation of the PDP-8 instruction set on a single large-scale integrated circuit chip), which could be adapted to running laboratory experiments with the modest temporal and computational requirements of the typical psychology experiment.

Imagine the efficiency of developing an experimental program on a fast Macintosh IICI or Quadra, then downloading the compiled code to a remote processor that could run the Macintosh operating system and free up the main computer for further development work. In a sense, of course, this is what we are doing when we develop an experimental program on a fast machine and store it on a server, from which it is retrieved over the AppleTalk network by Mac Plusses that actually run the experiments. The Mac Plus, configured as the intelligent peripheral of a Macintosh IICI, is just another instantiation of the Wheel of Reincarnation.

The Economic and Social Dimensions of Computing

Certain economic and social dimensions of the widespread use of computing are unlikely to be broached except in the unusual context of a presidential address. One is that even our day-to-day productivity management uses of the computer must substitute in some measure for more traditional ways of achieving academic and scientific work. There are personal productivity benefits, of course, in accuracy, speed, and the range of things that are possible. Part of the productivity that the computer affords is unique. No institution could, without computers, afford the resources and flexibility that make it possible for the first hard copy of a classroom handout to be gener-

ated just minutes before the class begins (or in some cases, a minute or two *after* it begins).

These advantages may have unforeseen costs, because part of the computer's productivity comes at the expense of investment in other human resources. Instead of hiring and training a research assistant who is knowledgeable about statistics to help with calculations (and who would, in the process, get some valuable experience), I may find it a more efficient use of my own time to crank out an analysis of variance in a few on-screen minutes. Because the start-up costs of involving students in my research are high, if I must first train the student to work with an unfamiliar programming language or package, I may decide to give the "smarts" to the computer, instead (it will not graduate and take its knowledge elsewhere), and give the student assistant a less involved role. I no longer need to rely on, or hire, someone else to type a grant application or class handout. The lettering pens and T-squares have been retired in favor of presentation graphics software, and the person once hired to draw graphs has had to seek a new career (as a software developer, perhaps). We employ fewer secretaries than we would need otherwise. Furthermore, to paraphrase a concern that John Castellan has eloquently expressed (1981, 1991), no one who lacks these personal productivity tools can realistically be competitive for grant support, or be in effective communication with the broader scholarly community over the electronic networks. The "have-nots" are still at risk.

Our view of the place of the computer in academic society is not the only possible one. A colleague at a nearby institution recently hosted a visiting scholar from China. She was demonstrating a spell-checking utility, and the Chinese visitor was sincerely impressed. But when my friend asked her, "Wouldn't you like to take one of these back to China with you?" the reply was immediate and emphatic: "Oh, no. Checking the spelling is someone's job!"

Another hidden social cost of our reliance on computers comes from the manufacturing process. Much of the computing hardware that we use, from individual chips to entire computers, is manufactured or assembled off shore. I know the wage rates for such work would be considered exploitive if they were imposed in our country. Off shore, as here, there may be environmental effects of manufacture. Recently, the regular decrease in certain atmospheric pollutants at the Grand Canyon each Monday and Tuesday has been correlated with the prior weekend's shutdown of the Los Angeles area's electronics industry, 500 miles to the west. I do not mean to imply that psychologists' use of computers contributes more than a minute amount to the social and environmental costs of our economy, but we should keep in mind that computing is one of many areas in our society where we are insulated from some of the social and economic consequences of our individual purchasing decisions.

At the other end of the computer's life cycle, there are also costs that are not typically taken into account when the equipment is ordered. When the Hamilton College Computer Center shifted from Terak computers (they are

graphics-enhanced LSI-11s) to Macintoshes in its word-processing lab, the psychology department adopted a number of the Teraks, to save them from the dumpster. They have been serviceable laboratory machines for several years. Since we have an increasing investment in Macintosh software and find the Macintosh environment more efficient for program development and more richly endowed with relevant software, we have tried to get rid of the old Teraks (except the few that have become "intelligent peripherals" for Macintoshes). Recently, one of our staff ran into his counterpart at Cornell University, and asked him if Cornell would be interested in buying a used Terak or two. The Cornell response was, "Would Hamilton like to buy fifty of them?" Hamilton still has a closet full of Teraks. The lifeless hulk of a Digital Vax 780 may moulder in the closet of more than one psychology department, because the cost of dismantling it can be applied more pragmatically to the purchase of a couple of Mac Classics for the graduate students. The absolute computing power of these idle Teraks and Vaxes is no less than it was on the day they were bought; it is only in comparison with current alternatives that they fall short.

Return to the Temporal Dimension

The past. Let us see how things looked 10 and 20 years ago. Castellán (1991) recently systematically reviewed trends in the content of *BRMIC*, so I'll take just two examples that particularly impressed me when they first appeared. In an invited address to the second annual meeting of the NCUOCP, Donald Norman predicted the eventual manufacture of the Dynabook, which would incorporate a small, integrated display, cassette tape reader, and keyboard, and which together would have about the same size as that of an encyclopedia volume (Norman, 1973). Norman foresaw that the Dynabook would have a fairly specific purpose (assistance with reading, note taking, and similar intellectual tasks). As the contemporary general-purpose laptop has been realized, these are only a small fraction of the actual uses. Norman could not have foreseen in 1973 the computational power that is now crammed into the laptop. The Mac Powerbook 170, at 6 pounds, with 4 MB of memory and a 40-MB hard disk, is about four times as powerful in speed and memory capacity as the IBM 360 mainframe that served the whole Cornell campus 20 years ago, which cost, taking into account inflation, five or six orders of magnitude more than the laptop.

As for what the 1973 dollar would buy, Norman reported: "Today, for around \$5,000 you can have a rather powerful machine." This "rather powerful" machine had a home television set for display, a cassette tape or one of the "new" floppy disks, and 4,096 words of 8-bit memory.

In a similar vein, Castellán, in 1980, noted that for \$2,500-\$3,000, one could put together a "viable system to do simple experiments" (Castellán, 1981). He had in mind a TRS-80 or an Apple II system, with a floppy drive and 64K of memory.

Conspicuously absent from both discussions was any mention of what software would be used for in these low-

end machines. Presumably the experimenter worked from ground zero to write a program in assembly language or an interpreted BASIC. There was neither the installed base of the same kind of machine, nor the media to support the wide distribution or exchange of special-purpose software. That is changing. Today, for between \$1,000 and \$2,000 (with educational discounts), one can buy a PC clone or low-end Macintosh with 2 MB memory, and for another \$500 or less, one can acquire one of several special-purpose laboratory-oriented software packages.

We now get a lot more for our money. What has really changed, however, is not hardware or cost. What has changed is our level of expectations, our notion of what Norman's "rather powerful" or Castellán's "viable" system must be able to do, how fast it must do it, and how much effort should be involved in getting to the point of actually running an experiment.

It is sometimes useful to reverse one's view, as if looking through the wrong end of a telescope, so let us look back as far as Norman looked forward. What would be necessary to get the equivalent computing power of a Macintosh II, using 1960s technology? Consider that the memory chip for a Macintosh holds 8 million bits of information in one twentieth the volume of a PDP-5 logic card, which had just three functional units. A few calculations on the back of an envelope will lead to the conclusion that a computer of 1991 capability constructed with 1967 components would look something like what is shown in Figure 5.

The present. What are the biggest challenges at present? I have already alluded to the general category of social costs, to which all computer users contribute in some degree. But some challenges specifically affect users in psychology, especially the evolving area of software development.

First, we are not very efficient in achieving appropriate economies of scale and communication across laboratories in developing software. Especially for the development and widespread distribution of materials for teaching, the mechanism that works for books (textbook publishers that support development, edit, aggressively distribute, and pay royalties to authors) is not adapting. This is unfamiliar turf for the companies, and the possibility of unauthorized software redistribution (piracy) threatens the return on publishers' investments. In addition, the evolution of computing platforms makes product lifetimes unpredictable. With major operating system upgrades every 2 or 3 years, one must run faster and faster to stay in the same place, owing to new development costs whenever there is a major operating system upgrade.

Second, the spectrum of resources available on different campuses continues to divide us into haves and have-nots. For Castellán, in 1980, the issue was access to electronic mail or Bitnet; now access to Internet services like FTP, Telnet, Finger, and NetNews defines full membership in the scholarly community.

Third, after the massive shift of computation from the Computer Center to our laboratories, the pendulum is swinging back to a central facility where economies of scale may be realized for services such as disk backup,

Figure 5. The hypothetical implementation of a contemporary laptop's computing power, using 1960s components. The Golden Gate Bridge indicates scale.

mail, and network access. What single user needs the most cost-effective disk, a 2 GB Wren drive, and the tape drive to adequately back it up? Individual users must buy disk space at retail at a high multiple of the optimal per-byte cost. The administrative challenge is to achieve a balance of support between local and central services (mail, printing, backups, maintenance of central storage, etc.) so as to fully exploit economies of scale. The further challenge in the present climate of fixed or diminishing support will be to make sure that all users are appropriately accountable for the resources that they use.

The future. In this rapidly evolving environment, it is foolhardy to make specific predictions about the future. Nevertheless, some general trends seem clear. While the pendulum has swung strongly toward giving users individual resources in personal computers and workstations, the return swing will favor tying resources together with even more flexible connections among machines and to networks. With Internet, I am now a virtual user of some machines (I can read my mail from any terminal on any campus), but I can't yet work on my desktop Macintosh from afar as if I were actually there.

Because we can access and modify information from many places, it is becoming important that in the system as a whole there be a single, authoritative representation of that information. It is now too easy to have multiple versions of a manuscript or a data set on the office machine, on a portable, on the home desktop, on the server, and elsewhere, and it is sometimes a challenge to identify and access the most current copy. A start toward the necessary integration is the Alias feature of Macintosh System 7.0, and the Remote Access program, which permit the same files and disks to be conveniently accessed

from many different machines, including dial-in access. In the future, I expect development of software tools that will automatically integrate work from multiple sources, which, of course, might be the members of an intercampus research group as well as the same person at different places.

The other trend that I expect will change the way in which we do our work will consist of programs writing programs—that is, programs that give users access to the same process at different levels of complexity, for different purposes. Multilevel representation of programs is

Figure 6. The tradeoff between the ease of programming in a computing language and the inherent power of that programming language. Both scales are highly arbitrary.

a feature of many compilers; they compile a high-level code to a symbolic assembly code that, in turn, is assembled. Representation of the same program at different levels of programming complexity helps mitigate the power-friendliness tradeoff (Figure 6). Higher level languages that are easy for novices do not have the power of more complex languages to implement complicated processes efficiently. However, using a complex lower level language requires a greater learning investment and possibly postponement of most research-relevant work.

In the Macintosh environment, we are coming to refer to languages that give access to the program at different levels as *scripting* languages. There is a high-level representation of a process, experiment, or computation, in an iconic or other metaphoric representation, that is easily programmable by the relative novice. The high-level user interface provides considerable hand-holding (automatic on-line help and a syntax constrained by the user interface). The Macintosh user interface standard is a good example of such a high-level representation: operations that are syntactically meaningless in a particular program state are ostentatiously disabled.

In a scripting language, in addition to the high-level representation of the process, there is a lower level representation of the same process, the script. A script is a description of a sequence of operations in a general-purpose programming language. The script is ordinarily represented in plain text, rather than iconically, and so it is directly editable. Thus, a script (once produced by the high-level interface) can be executed, modified, or saved. Because the ideal script language embodies a general-purpose programming language, the script can be enhanced to include sequences of operation or contingencies that the designers of the high-level interface did not foresee. As a consequence, the scripting language may combine the ease of use of an icon-based language with the power of a general-purpose language. Once a program is generated with the use of the iconic user interface, the saved script provides an archival record of the process, which can be executed at a later time, to exactly replicate a prior analysis, or be modified to perform a related analysis, or be shipped to another machine for execution or be mailed electronically to a colleague at another institution.

A script might even be generated by another program entirely. Imagine an experiment-control program that itself had no statistical capabilities, but that generated a script for the appropriate statistical analysis as a by-product of each experiment, and automatically submitted both script and data to a statistical package for final analysis.

The final advantage of a script-producing high-level application program is pedagogical: by studying the script

produced for different processes, the new user can both experiment and learn by example. Many Macintosh programs have a macro capability that approaches the kind of scripting capabilities that I am describing here, but few have profoundly exploited it yet.

The Long View

I would like to close by looking at time on the human scale. Whenever I think about the pace of change in our world, of which the evolution of computers is just a part, I cannot help but think about the life of my maternal grandfather, J. Howard Cloud. He was born around 1870. When he dropped out of high school, he was hustled off to medical school, which will give you an idea of what unsophisticated tools medicine then had at its disposal. Dr. Cloud began his career as a horse-and-buggy country doctor. He would ride out in the starry cold night of the Pennsylvania coal fields to deliver a baby or comfort the family of a diphtheria case. Exhausted, he would sleep in the buggy on the ride home. The horse knew the route, and it would stop at the grade crossings and wait for my grandfather to wake up and look for trains before proceeding. (Today, we might characterize the horse as an organic expert system).

Over my grandfather's long professional life, he came to rely on antibiotics, insulin, diagnostic X rays, and the rest of the powerful armamentarium of modern medicine. There were equally miraculous changes in his nonprofessional life. On those buggy rides home, he told me, he would sometimes look up at the moon and wonder about the mysteries of its far side, forever invisible to us on earth. Near the end of his life, he was captivated by the expeditions that sent representatives of our species to explore that moon's mysteries in person. All this in one lifetime.

Two decades of using computers in psychology is the equivalent, on the human time scale, of my grandfather's age when he first rode out in his medical practice. The changes that are in store for us in the decades to come will be every bit as miraculous as they were for my grandfather.

REFERENCES

- BELL, C. G., MUDGE, J. C., & McNAMARA, J. E. (1978). *Computer engineering*. Maynard, MA: Digital Press.
- CASTELLAN, N. J., JR. (1981). On-line computers in psychology: The last 10 years, the next 10 years—The challenge and the promise. *Behavior Research Methods & Instrumentation*, 13, 91-96.
- CASTELLAN, N. J., JR. (1991). Computers and computing in psychology: Twenty years of progress and still a bright future. *Behavior Research Methods, Instruments, & Computers*, 23, 106-108.
- MYER, T. H., & SUTHERLAND, I. E. (1968). On the design of display processors. *Communications of the ACM*, 11, 410-414.
- NORMAN, D. A. (1973). The computer in your briefcase. *Behavior Research Methods & Instrumentation*, 5, 83-87.