Northeast Regional Data Center

/Update

Volume 19, Nos. 5 & 6, May/June 1991

Teaching the Computer: A Decade of Dazzle

University of Florida President Dr. John Lombardi presented this speech at the Florida Educational Technology Conference earlier this year.

In the Beginning

What a decade it has been! We began the microcomputer wars with a battle between the Radio Shack TRS-80 and the Apple II, and today, here we are with Tandys, Apples, and IBMs, struggling for supremacy over the minds, hearts, and dollars of our classrooms.

Back in the dark ages of educational computing, in the late 1970s and early 1980s, we participated in a movement of evangelistic fervor.

The campaign to put computers in the schools mobilized us to convert the heathen to computer literacy and to spread the gospel of our favorite machine. We fought bitterly over black-and-white or color displays and the rigidity of the TRS-80s or the flexibility of the Apples. We heaped scorn on those who didn't see the righteousness of our way, but we reserved our deepest feelings for the stubborn teachers and school administrators who didn't see the dawning of the new computer age. Never mind that most software converted trivial print media, that the complex, cranky hardware proved unreliable, and that games represented the only truly good stuff available; we who carried that computing torch knew that we were on to something big.

In those days, we served on parent-teacher committees to advise school boards on what to buy for the classroom. We argued about whether we should put the machines in a computer lab or disperse them into every classroom. We discussed

continued on page 3

Use Rice Mail in CMS

The Rice Mail system from Rice University is now the only mail user agent supported by NERDC under CMS. The "NERDC" mail interface and the EMAIL command were removed from the system on May 1, 1991.

Two manuals that document Rice Mail are available to you. You can print a copy of Rice University's "A User's Guide to Electronic Mail" by entering MANUALS in CMS and choosing to print the manual MANUAL/MAILBOOK. (This manual must be printed on wide paper, FORMS 3001.)

You can obtain UF Professor Dick Elnicki's "E-mail Starter & More" from the NERDC Support Desk. This manual contains information on email specific to NERDC users including sign on information and a "quick and simple" guide to sending email.

For more information about using electronic mail at NERDC, contact the CIRCA Consultant at (904) 392-0906, SUNCOM 622-0906, or the NERDC Support Desk at (904) 392-2061, SUNCOM 622-2061, CONSULT@NERVM. ✣

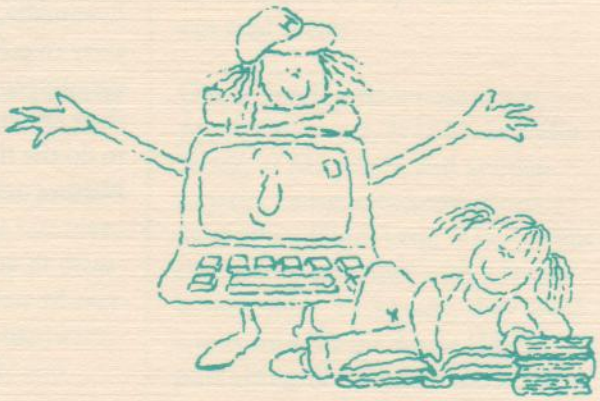
In this issue...

continued from page 1

in-service training for teachers and we worried about how to make the students computer literate. We fussed with questions of policy: should we allow instructional computers to be used for mere utilitarian purposes - such as grade books, typing, and accounting- or should those precious Apple IIs be kept pure, occupied by educational tasks? We struggled with the problem of software piracy, our high-minded principles often defeated by destructive copy protection schemes, idiotic educational purchasing policies, and the ingenuity of our students. We argued over whether students should be introduced into the computing faith by using applications or by understanding programming. Over time, events and technology answered many of our questions and invented new ones.

As an amateur microcomputer enthusiast, the age of my children and the circumstances of my life determined my focus on educational computing. By the accident of age, my children suffered the consequences of being in primary and secondary schools during the birth of the microcomputer age. Like any good evangelist, I worked first to convert my own. Both children received the gospel of microcomputing without mercy. They learned how to program in BASIC. They did science fair projects on the family Apple. They tested every piece of semi-useful educational software I could find. They even endured various versions of typing tutor. Long suffering and of mild temperament, my children, recognizing the misguided good will that underlay their father's monomania, went along and did most of what I asked. Being young and wise beyond their years, they quickly solved the computing problems I thought were substantial and went on to other interests. At first, secure in my evangelical pursuit of universal microcomputer literacy, I assumed that I had failed with my own children and began to doubt my faith. As time passed, I came to recognize that they, and many others, had the right idea.

My children refuse to think of computers as anything special. For me, the microcomputer remains a marvel of technology, a tool of extraordinary power, a magical box that multiplies my abilities and enhances my work. For them, it's just another magical device in a world full of magical devices. Where I marveled at what the machine could do, they wondered why it couldn't do more or do it better. They quickly saw that they could turn a book's pages faster and easier by hand than they could turn them on the screen, and they could see the book's words more clearly. So they avoided the educational software I brought home, unless I stood there. Then, to keep old Dad happy, they would go



continued from previous page

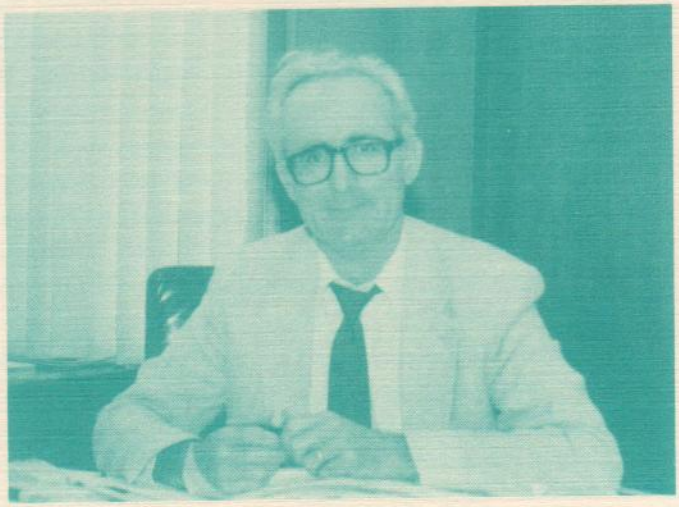
through it. They quickly discovered that typing tutors were just as boring as any other means of teaching typing, so they quit that. They found the Apple II's primitive word processing programs more trouble than they were worth. And they recognized immediately that programming was just another difficult task, not worth the trouble unless you liked doing it, which they didn't. They did discover that the really good Apple games tested their skills and were fun. So they played them all, traded copies, and conspired with me to break copy protection schemes so they could illegally trade game software with their friends (all evidence of which no longer exists).

Eventually, they grew up and went away to college. Both kids have computers, hand-me-downs from their father. They use them as tools to do the things they want to do. Neither one wants to know how they work. Both expect the computer to do its thing for them. When the computer doesn't work, they call me.

While I went on to other things, adopted other evangelical enthusiasms in education, you and your colleagues have been working on that wonderful machine, building better software and hardware, until today, over a decade later, I can find in the College of Education on our campus, machines and material that indicate that much has happened to bring the promise of useful computing for the classroom closer to coming true.

Lombardi's Laws

As we continue this on-going romance with the thinking machine, some general principles appear to hold true about computers and education. Let's call them Lombardi's Laws.



University of Florida President Dr. John Lombardi

I—The Optimal Machine With Its Software Will Always Cost Too Much

I discovered this law rather early in my computing experiments. I read in the computer magazines that the cost of computing was declining rapidly. Memory, disk space, computing speed, color monitors, mice, and all the other hardware, they told me had declined by half, a third, two-thirds or whatever, in the last two, five, or ten years. They published charts and graphs indicating the declining cost per micro-megaflop of computing. Yet, when I checked my computing savings account, it remained relentlessly empty. I'm now on computer number four (not counting laptops). My first machine cost $2400 dollars. My last machine cost $2400. The two machines in the middle cost $2400 each.

Over the years, software has become much... easier to use, but many of these benefits require the purchase of ever more powerful and fancy hardware.

I followed my normal procedure and bought each machine just after the introduction of a newer and more powerful model, thereby guaranteeing that my machine would work since it wasn't a new design. Each machine was much more powerful than the machine before, so I did indeed get more micro-megaflops per dollar, but I still ended up paying $2400 dollars for each machine. This wonderful, terrific pace of change that brought the price down dramatically per micro-megaflop made it necessary for me to buy four machines before the first one wore out.

Since schools keep their machines even longer than I do, it's clear that the improvements in hardware power do not necessarily translate into less money required to equip individual schools and teachers with computers. Sometimes the improvements in cost per micro-megaflop result in higher costs for schools whose still usable machines no longer can be used.

II—Software Expands To Fill The Hardware Available

Over the years, software has become much better, slicker, more intuitive, and easier to use, but many of these benefits require the purchase of ever more powerful and fancy hardware. As the software gets fancier and better, it doesn't get fundamentally cheaper. While the economics of software production do not match those of hardware, the power and effectiveness of current, state-of-the-art instructional software commands prices that run into the hundreds of dollars per copy. A long way from the $29.95 programs I used to buy for my Apple II that my children's schools couldn't afford.

III—Incompatibility Guarantees Profits

Intentional incompatibility separates the printed word, the ubiquitous book, from the hardware-software computing machine. The book is a standardized educational item. No matter who publishes it or writes it, no matter how it looks, whether with or without pictures, with or without color, or of large or small format, you, me, our kids, and our kids' kids can read it. I can go to the library and get a book published in 1891 and read it just as well as a book published in 1991. This is so because Gutenberg, the fool, didn't know about copyrights and patents. Consequently, everyone can make books the same way, and all of us can read them.

Our technological innovators at Apple Computer, at IBM, at Radio Shack, at Microsoft, and at other superior companies know much better than Gutenberg. They invent computing machines with a lawyer at their side. To be sure, they made a mistake with the Apple II by publishing all its source code and specs for anyone to see and use. But with the Macintosh they did it right. Proprietary code, proprietary operating systems, a wall of patents and copyrights, each designed to be sure no one could make products that involved the Apple computing machine without paying large sums to Apple. Good business sense that made Apple a lot of money.

continued

Up Front

continued from previous page

IBM tried the same thing, but coming later than Apple, they couldn't quite get it done. IBM became more open and more accessible to lots of people, much like the Apple II. Thanks to this business error, IBM gathered lots of what bean counters call market share. Yet neither Apple nor IBM, even together, can approach the 99.9% market share of Gutenberg's book.

Who cares? You and I care, because Apple and IBM don't talk to each other. That means you and I have to pay for everything twice. We pay once when Apple invents something that won't run on an IBM. We pay again when IBM reinvents the same thing differently so it can run on their machine. But the greed of the vendors doesn't stop there. Not only do they make IBM and Apple incompatible, but they then make the subsequent versions of their own machines incompatible with earlier versions. Each incompatibility fragmentation of the market generates higher profits to the company and extra costs to the schools and the consumers.

Were this not enough, they then tell us, "Now, you have our incompatible machine and you have our proprietary software, but you must not use the software you bought on more than one of our machines at a time." Why didn't Gutenberg think of that? Imagine, you go to the store and buy a paperback copy of the latest best seller, you read it, you pass it to someone else who reads it, you take it to the used book store and trade it for another paperback. Sometimes, you read part and your spouse reads another part of the same book. On occasion, with children's books, you and your child apply two different sets of eyes to the same print at the same time. No modern technology company would permit this failure of good business sense. That's why Gutenberg is famous but died poor.

Did you ever read the fine print on the user license packed with your software? You can't lend it, you can't install it on more than one computing machine at a time, you can't copy it, you can't sell it, and in fact, although you think you bought it, you didn't, you only bought the right to use it as defined by the license. When you get through using it their way, you have to give it back to them or destroy it.

I'm told that someone has to protect the companies whose revenues are reinvested in better software for us all. You may have a vision of skinny hardworking nerds in the basement of a grim concrete block building drinking Coca-Cola and eating Twinkies as they grind out new educational software. You should visit the headquarters of a real company like Microsoft or Apple Computer. There, you'll find conditions so elegant and refined you'll think you've stumbled on a brokerage house or a trendy lawyer's office. You'll find employees

FYI...

The */Update* mailing list has been purged. Thanks for renewing your subscription. We hope you enjoy your next two years of */Update*.

parking their expensive cars (costing more than your school district's budget for software) under the trees near quiet, air conditioned, and beautifully landscaped buildings. As you speak with them in hushed tones about the computing needs of school children in disadvantaged neighborhoods, it may occur to you that possibly all the artificial incompatibilities between software and hardware products do not serve the best interests of the children or the teachers.

IV—Software Is Much More Important Than Hardware

Everyone of us who has participated in any part of the educational computing world knows that the computing machine makes no difference if the software doesn't do the job. However much we may read tech specs, hardware has no utility without software, except as door stops. Software is much harder to do than hardware, which is why hardware advances in six-month intervals and software takes years. Hardware involves taking a single thing and making it bigger, smaller, or faster. It piles similar widgets on top of other widgets, cramming them into smaller and smaller spaces for the same price. This is tricky, it takes lots of smart people to do it, but it isn't exactly hard.

Software recreates bits and pieces of human understanding in an inanimate medium and delivers it to unknown audiences with unpredictable expectations and abilities.

Software imitates life, and life, being incredibly varied and often unpredictable, resists imitation. Thus, software takes more time and more creativity than hardware.

V—Computers Do Simple Things In Complicated Ways Better Than They Do Complicated Things In Simple Ways

This principle works especially well with educational applications. When we tried to make the computer substitute for the routine stuff of exercise books, drills, and other repetitive and relatively simple things, we found that the cost and programming effort of doing it well greatly exceeded the cost of duplicating drill sheets or handing out exercise books. Moreover, the programs themselves often were more complicated to use than a piece of paper and pencil. We also discovered that the truly sophisticated applications, such as simulations of historical reality, required an awesome investment of time and resources to develop and an expensive computing environment to deliver. At the same time our students found the results cute but boring, and our colleagues found fault with the simulation's content and logic.

Teachers, Students, and Computers

Lombardi's Laws have the virtue of predicting the past, the fundamental purpose of all rational systems. If we look at what's out there on the cutting edge of software and hardware development we find that these laws also work well. Because the cost of the basic effective computer workstation remains stable, and because state-of-the-art applications often require double the cost of a basic workstation, the illusion we once held that computers would become a child appliance has fundamentally dissipated for most school systems. Instead we

continued

continued from previous page

have computer labs and instructor workstations.

Some might regret this result, but I'm not so sure. While a confessed and unrepentant techno-junkie, the theory that computing machines will replace teachers has never much impressed me, perhaps because I'm a teacher. Some of you are old enough to remember that television was supposed to do that trick some years ago, replacing us, the regular good teachers with an exquisite master teacher, reaching out to the masses of students through television. We tried it in the universities where it survives today in vestigial form in business and engineering, but not in the core disciplines of the liberal arts and sciences.

Computers are much the same. People who think computers can replace teachers also think that libraries can replace schools and that commercial television can replace conversation. Computers, libraries, and television are tools in support of education but they do not replace teachers. Teaching, that elusive and difficult art of translating the substance of a discipline into a frame of reference for further learning, is not an information-transfer mechanism. Teaching uses information to explain a process of thought, teaching requires information to inform judgment, teaching explains the proper use of tools for understanding complicated things. Once we recognize that information and tools do not teach, but assist learning, then we can better understand how computing machines contribute to teaching and learning.

I can remember trying to persuade a fine teacher in an elementary school that she should be working to get computers in the hands of every student. She, with the charm and grace of an experienced and diplomatic teacher coping with an enthusiastic, well-meaning, but not too bright parent, agreed but told me we should first get the tools in the hands of the teachers. Without the teachers, she said, the learning can't happen. She was right, as my children and others have demonstrated over the years. The results to date have been some exceptional applications emerging in support of teaching that conform to Lombardi's Laws as I recently observed in some examples at the University of Florida College of Education.

As a humanist, I've always assumed that the best use of computers would be in the development of science-based applications with a known universe, relatively simple rules, and predictable relationships. Sure enough, an instructor can now demonstrate, cleanly and effectively on screen for the whole class to see, the principles of mechanical physics. Levers move, balls roll down inclined planes or fall from heights, springs compress and stretch, pendulums swing, and everything changes at the inspiration of the instructor. If the students don't

Attention All Departments

Please remember to renew your encumbrance/purchase order or close your account before the end of the fiscal year (June 30 for many agencies). You must notify NERDC Accounting in writing if you plan to close your account.

understand the relationship between the angle of the inclined plane and the speed of the rolling ball, we can change the angle of the plane in seconds and re-run the experiment. If the students' questions indicate that we didn't design the first experiment successfully, we can redo the experiment instantly, finding just the right combination of illustrations to capture the student imagination. This nifty application takes relatively modest hardware and software, plus the gadget that makes a computer screen appear as if it were an overhead projector slide.

With a good bit more money, you can enter the world of audio, video, computing. This is the new wave that will raise the cost of computing hardware so that advances in miniaturization and computing power will not reduce the basic workstation price. Laser disks, video disks, voice input-output, music synthesizers, various forms of video tape, all offer the possibility of expanding the range of effects and materials available to the instructor.

We can now get a little program that will put the entire collection of paintings from the Louvre or the National Gallery in Washington on screen. You can view paintings by chronology, artist, style, or subject, for example. You can teach art appreciation much as your colleague teaches physics, showing examples, reordering them quickly in the event those first chosen don't work with your class. Your tools for teaching have, once again, expanded.

In the End

What does all this tell us about computing? First, it tells us that there is no such thing as educational computing. Instead there are a host of tools, applications, appliances, and other devices that use computing technology. Computing is not a thing, not a religion, not even an industry. It is a tool.

Second, tools are only as good as their design and only as useful as their function. If I want to fix my truck, it does me no good to have the most elegant and wonderful gardening tools. The same holds for computing tools. You want to teach math? Don't get a word processing program. You want to teach writing? Forget about spreadsheets.

Computing tools serve teaching. They neither define teaching nor accomplish instruction.

Third, computing tools serve teaching. They neither define teaching nor accomplish instruction. Teachers do that. If you have nothing to teach, the fancy hardware-software multimedia presentation of the National Gallery of Art remains just a clever curiosity. In the hands of a great teacher, tools enhance learning. In the hands of the uninformed, great tools become expensive gadgets. Computing tools enhance the effectiveness of good teachers, and, adequately prepared, students learn more quickly with access to these tools.

Fourth, computer manufacturers and software developers who resist basic standards, insist on proprietary systems, and encourage obsolescence are the enemies of quality instruction, for they artificially raise the price of innovation, putting many good tools out of the reach of teachers or students.

Fifth, teachers - not administrators, not software developers, not hardware designers, and not enthusiastic parents - provide the final test of educational computing. If the stuff is good, if it works easily, if it is better than the materials we now have, and if the school can afford it, then the teachers will adopt it. If it is hard to use, requires more preparation time than current methods, produces similar educational results, and costs too much, then the teachers will resist it.

* * * * * * * *

If there is a moral to my story it is this: Teaching and learning require knowledge, inspiration, commitment, time, and resources. Computing gives yet another set of tools to help with teaching and learning, but these tools, however fancy, elegant, and fun, cannot substitute for the quality of the teacher and the commitment of the student. ✣