5-1990

The Gears of Childhood

Glen L. Bull
University of Virginia

Gina Bull

Judi Harris
College of William and Mary

Recommended Citation

Bull, G., Bull, G., & Harris, J. (1990). The gears of childhood. Logo Exchange, 8(9), 11-16.

The Gears of Childhood

by Glen Bull, Gina Bull, and Judi Harris

In his foreword to *Mindstorms*, Seymour Papert describes his childhood fascination with gears. At an early age he became intrigued by the movement ratios produced by rotating gear combinations of different sizes. Gear ratios became the method which he used to conceptualize the multiplication tables, and later differential gears became the model for building equations with two variables, such as $3x + 4y = 10$.

In Papert's terminology, gears served as a transitional object. They provided a concrete object that he understood which could be used as the basis for building active experiential understanding of a new abstraction: mathematical equations. Papert proposed that computers could be used similarly to provide inexpensive, flexible transitional objects. In essence, computers could be used to create "electronic gears." An electronic turtle on the computer screen could serve as a transitional object because children could relate the movement of the turtle on the screen to the movement of their own bodies in space. This might provide an accessible and motivating method for exploration of geometry and other mathematical concepts.

At the time that *Mindstorms* was written, this was a rather radical way of thinking about using the computer in education. Previously, the chief tendency had been to think of the computer as a mechanized replacement for the teacher, rather than as a tool that could be used by the teacher or learner as a bridge to other concepts.

Over the years, Logo and turtle geometry provided a rather durable transitional object. When Logo first experienced tremendous popularity, there arose a debate as to whether Logo was the only valid transitional object that could exist on the computer. This might be termed the phase of *Logo as religion.*

Enter: the Heretics

At about the same time, Bob Tinker was evolving the concept of microcomputer-based laboratories (MBL) at the Technical Education Research Centers (TERC). The concept of MBL is that rather than using the computer to *simulate* scientific experiments, probes and sensors attached to the computer could be used to collect actual data to conduct real scientific experiments. Many of these ideas were described in a column with the wonderful title of "Tinker's Toys" which he wrote for TERC's *Hands On!* newsletter. In one of these columns Bob noted that for some MBL applications Logo lacks the speed required to acquire some types of data in real time.

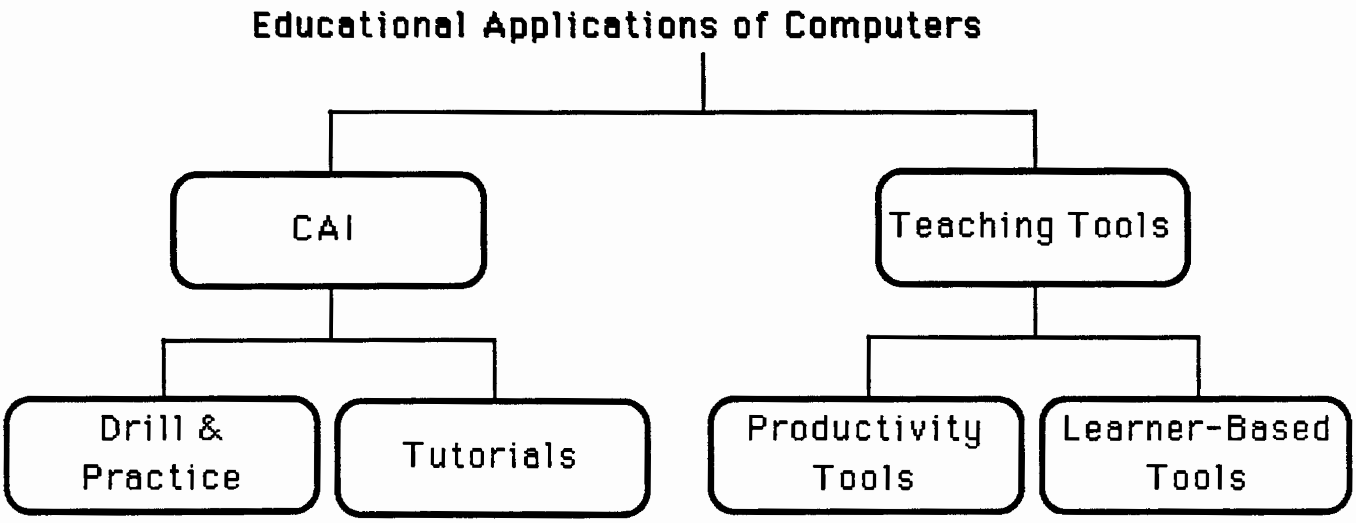
This was hardly a startling observation, since Logo is an interpreted language with moderate though not blinding speed when implemented on an eight-bit microprocessor. However, at the planning meeting for the first national Logo conference, this revelation was treated as pure heresy by some. In the midst of an active discussion, Hal Abelson (of M.I.T.) finally said, "Bob Tinker's a pretty good guy. If he's making these observations, maybe we should be listening to what he's saying."

In response to the discussion that followed, Abelson said, "Logo is not the only way." When challenged to list other approaches, Abelson thought for a moment, and then said, "Well, Boxer is a possible model." Boxer is a programming language inspired by Logo that uses windows (or boxes, hence the name "Boxer") as containers for procedures. At present Boxer requires the power of a Sun workstation with several megabytes of memory to run. But then again, when Logo was first developed, microcomputers had not even been conceived, and hence Logo was not generally accessible in the public schools either. Readers interested in the characteristics of Boxer can refer to the citation listed at the end of this article.

Hal Abelson probably did not expect casual comments made over coffee to be remembered nearly a decade later, but his remarks get at the essence of an important issue. "Is Logo the only valid transitional object, or are there other approaches?" Before considering this question, we would like to turn the clock forward a few years to a discussion held with Tim Riordon while walking on an Atlantic beach. At that time, we were considering the phenomenon of *Logo as dogma.*

Dogma in this case refers to blind adherence to an edict without a sensitivity to reasons underlying what may have been intended as a guideline rather than a stricture. Recursion is a good case in point. The capability for recursion is one of the more desirable characteristics of Logo. Thousands of words have been written about recursion in Logo. Some have interpreted this to mean that recursion should always be used and that loops should never be used. In point of fact, there are instances in which a loop is more efficient than a recursive procedure. For example, a loop may be appropriate in a condition in which an external switch or sensor attached to the computer is polled.

As we discussed this, we were bemused by the vehemence with which some at that time attacked the use of any non-recursive procedure, and spontaneously invented the concept of the *Logo police*. The Logo police, we decided, will monitor your behavior and break down your door at night and



carry your computer away if you are caught using a non-recursive procedure, or engaging in any other form of un-Logolike behavior. Although the Logo police are a myth (as far as we know), at times it does seem as though the rigid application of dogma stifled the very creativity and innovation that Logo was designed to express.

CAI and Learner-Based Tools

An important aid to our thinking about this issue soon emerged from the Technical Education Research Centers. This aid was the notion of *learner-centered software* (Mokros and Russell, 1986; Russell, 1986). At that time Susan Jo Russell, a researcher in special education, was concerned about the almost exclusive use of Computer Assisted Instruction (CAI) applications in the field of special education. She and her colleagues completed a national survey and found that at that time almost all uses of the computer in special education classrooms consisted of CAI applications such as drill-and-practice activities. They found that although teachers expressed interest in more open-ended uses of the computer, such as Logo, they found it difficult to measure their effects, and therefore difficult to justify their use. Russell and company suggested ways in which teachers could extend their own notions of instructional objectives to include goals for learning which were less content-specific, more process-oriented, yet just as documentable as those for CAI. Russell (1986) took the opportunity to point out that computers had the potential to cause teachers to reconsider not only how they were teaching, but also what they were teaching.

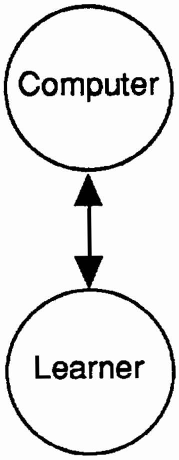
Russell and other researches at TERC described learner-centered software as an alternative to CAI which "has particular pedagogical characteristics which place more cognitive control in the hands of the learner" (Mokros and Russell, 1986, p. 185). For several years we have been using a modification of their term: *learner-based tools*. By this we mean software that is open-ended (having a variety of flexible outcomes) and learner-centered. Generally such software is distinguished from "utilities" or "productivity tools" by the potential for exploration or discovery. Logo is an example of a learner-based tool. Other early examples, we realize in retrospect, were attractive to us because they seemed Logo-like when they first came to our attention. They include Bob Tinker's micro-based labs as well as the talking word processing software developed by Teresa Rosegrant (Rosegrant and Cooper, 1985;1987.)

Interactive Structures

Good CAI programs can be very effective teaching devices. Unfortunately, good CAI can be expensive to produce. It has been estimated that it may take 200 hours of development to produce one hour of effective CAI. At that rate, one year of work is required to produce just ten hours of CAI. Another problem can be that the program may not present the information as the teacher would have preferred.

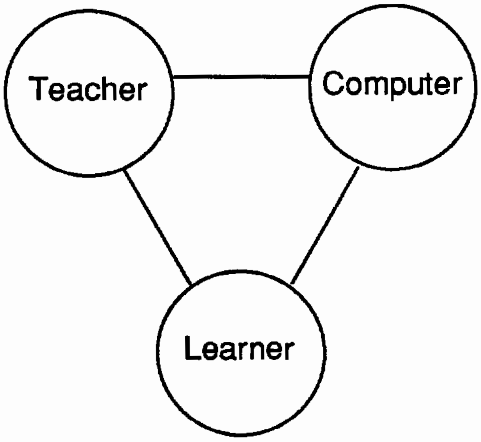
In traditional CAI, the computer replaces the teacher to some extent. In a drill-and-practice application, the computer drills the student on facts that have been taught by the teacher. In a tutorial program, the computer actually teaches the fact as well as directs and evaluates the student's practice efforts. There is a two-way interaction that occurs between learners and computers when students use CAI.

Two-Way Interaction
Computer-Assisted Instruction



In contrast, effective use of a learner-based tool requires a three-way interaction between the teacher, the computer, and the learner (Bull, Cochran, and Snell, 1988; Bull, Lough, and Cochran, 1987; Cochran and Bull, 1985). Logo is a good example of this type of interaction. The turtle is used by the teacher to illustrate the relationship of a new concept to existing knowledge.

Three-Way Interaction
Teaching Tools



Once the learner understands the relationship, the student can use the computer environment to explore the terrain. However, it is necessary for the teacher to provide "nudges" from time to time that edge the student into arenas that are apt to lead to productive discovery. One of the frequent misconceptions about this type of learning is that it is a "hands-off" process in which the teacher allows the student to discover concepts independently and rather randomly. It is to be hoped that discoveries of these types will occur, but facilitation by the teacher is a necessary and integral part of the process.

There are several ways to distinguish CAI from learner-based tools in the classroom. One is to ask whether a two-way or three-way interaction is occurring. Another important way is to determine whether the application is extensible. In some respects, a good CAI program is like a cleverly designed maze. If the program has been well-designed, the user will never be aware that there are boundaries beyond which it is not possible to go. However, CAI programs are of necessity finite with fixed boundaries. (It is possible that the area of artificial intelligence (AI) will change this, but AI applications are unlikely to affect public education in this century.)

In contrast, learner-based tools are always *extensible.* This means that the user can create uses and applications of the tool that were not envisioned by the developer. The extensible quality of these tools shifts the locus of control to the learner, which accounts for the derivation of the term *learner-based* tool. Logo is one such tool. Are there others?

Building a Learner-Based Toolkit

We suggest that *Boxer*, and certain uses of programs such as *Talking TextWriter*, as they were envisioned by Teresa Rosegrant, qualify as learner-based tools. Moreover, many hypermedia applications, as they have evolved over the last decade, could meet the criteria for this type of tool, depending on their use. In a hypermedia system, learners can move within a universe of knowledge, creating trailmarks and links as they go. Because the user directs the direction of travel, the locus of control is with the learner. It is true that the body of knowledge is finite. However, the 650 megabytes of information that can be placed on a CD-ROM (for example) constitute a rather large conceptual universe. For purposes of comparison, it can be noted that everything that Shakespeare wrote will fit in 7 to 8 megabytes of space.

HyperCard, written by Bill Atkinson, popularized the current interest in hypermedia programs, and now numerous hypermedia programs are available on almost all brands of computers. Bill Atkinson acknowledges that Logo was one of the inspirations that he drew upon as he was developing *HyperCard*. *HyperCard* extends the concept of "object-oriented programming" (OOP) which is found to a lesser extent in Logo. Turtles and sprites are examples of objects that can be programmed in Logo. Once the turtle is assigned characteristics (heading, pen color, etc.) it maintains those characteristics until they are changed. Some types of sprites even can be assigned a velocity so that they stay in motion until told to stop. *HyperCard* has many more objects that can be programmed in this way, serving as actors in a "script" created by the user. Certainly it is possible to do many of the same kinds of exploratory, extensible, learner-centered activities in *HyperCard* as in Logo. Hence, we believe it qualifies for the term "learner-based tool."

These types of similarities aside, we have found that teachers who use Logo can quickly develop similar teaching tools with *HyperCard*. Elsewhere in this issue of *Logo Exchange*, an article even describes a means of creating of a *HyperCard* turtle, illustrating the range and flexibility of this environment. Naturally Logo and *HyperCard* each have certain strengths that are not present in the other; they are not clones. However, we believe that the uses to which these programs can be put are similar enough that they should be placed in the same phylum.

Shall We Concentrate Upon the Tools or Their Use?

Learner-based hypermedia applications share another important characteristic with Logo. Although the learner may have almost unlimited directions to travel, it is likely that the journey will be facilitated by the presence of a guide who has previously covered the terrain.

The issue of interest here is not so much the tools themselves but their actual use. It is possible to write a drill-and-practice or tutorial program with both Logo and hypermedia programs. In fact, many teachers' initial explorations of Logo often involve development of tutorial programs. Some mathematics books for elementary grades contain examples of drill-and-practice programs written in Logo. Therefore it might be more accurate to say that Logo can be used as a learner-based tool than to say that Logo itself is such a tool.

The question we raise is whether the issues discussed in the Logo Exchange should have to do only with the programming language Logo, or a philosophy of teaching. We believe that the *Logo Exchange* is more about a pedagogic philosophy than about programming methods in a specific language. If Papert had focused on the specific transitional object of gears rather than the larger educational issues, he might have spent a lifetime refining more and more sophisticated gear systems that could be brought into the classroom as educational toys. This could have led to classics such as *Gear Storms: Children, Cogs, and Powerful Machines*.

In the early part of this century many carriage manufacturers defined their task as production of horse buggies and subsequently went out of business. Others defined their business as transportation and survived. This is why every car made by General Motors has "body by Fisher" (a large carriage manufacturer that survived) embossed on a plate on the door frame. At this juncture the issue of whether we are more interested in Logo as a programming language or Logo as an approach to teaching with computers is a crucial one.

Logo's Evolutionary Pattern

During the 1970s an almost unnoticed revolution occurred in the field of paleontology. This revolution, described by Stephen Jay Gould in his book, *Wonderful Life: The Burgess Shale and the Nature of History*, resulted in a reexamination of the history of life, including our own evolution. Gould describes the shift in thinking that occurred in the following way,

> ... in an error that I call 'life's little joke' (Gould, 1987), we are virtually compelled to the stunning mistake of citing unsuccessful lineages as classic 'textbook cases' of 'evolution.' We do this because we try to extract a single line of advance from the true topology of copious branching. In this misguided effort, we are inevitably drawn to bushes so near the brink of total annihilation that they retain only one surviving twig. We then view this twig as the acme of upward achievement, rather than the probable last gasp of a richer ancestry. (Gould, 1989, p. 35)

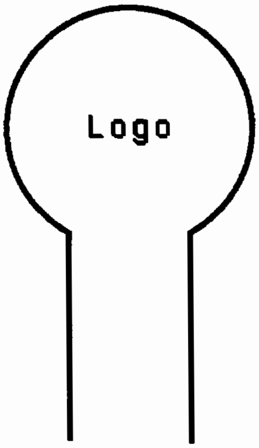
Gould notes that the view of evolution as an inexorable march of progress, culminating with the person telling the story at the peak of the evolutionary ladder, is an appealing one. However, he observes that the real success stories of mammalian evolution–such as bats, antelopes, and rodents–are the ones which present us "with thousands of twigs on a vigorous bush." Would it be more productive to consider Logo as the terminating event in the evolution of educational software, or should it be considered as one branch in a broader lineage?

One perspective is that Logo lies at the peak of a software evolution, unrelated to any other lineages. Rather than thinking of Logo as the end of an evolutionary line, we find it more productive to think of it as one of many examples of a thriving lineage of learner-based tools. Since Logo was one of the first of these educational tools, we will always have a strong interest in its use. However, there are now many interesting companion tools to explore as well. Thus we move from the *Logo as Religion* through *Logo as Dogma*, to the *Logo as Exemplar* phase.

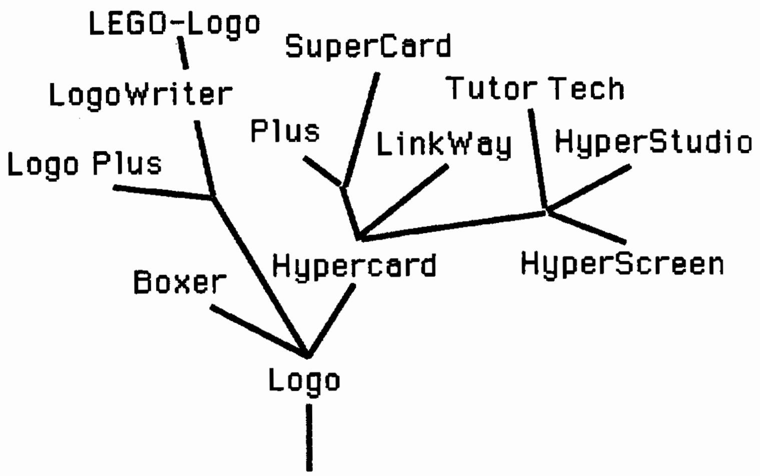
A Proposal: Logo as Exemplar

To signal the extension of a welcome to discussing all types of Logo-like tools, we propose a change in title of *Logo*

Electronic Cul de Sac?



Or Evolving Tools?



Exchange magazine to *Technology and Teaching Tools* (or *Journal of Learner-Based Tools*, or a similar, open-ended title.) When Tom Lough founded the *Logo Exchange*, Logo was the only example of this new instructional use of the microcomputer. Now that it has been joined by many other companions, we think it desirable to welcome them into the fold as well. Is Logo a transitional object leading to new and ever more interesting educational applications ranging from *Boxer* to *HyperCard*, or is it an electronic cul de sac? At one time it was not uncommon for many teachers to employ Logo as the sole tool for all applications from word processing to computer art. Now teachers have a much wider instructional computing tool kit. Rather than ignoring these small mammals (which have been busily eating dinosaur eggs) in the instructional computing community, we would like to extend a welcome.

In the seventies Logo was in its infancy. In the eighties it entered its adolescence. In the nineties it approaches adulthood. Papert used the gears of his childhood as a transitional object, which was used to found a new philosophy of educational computation. For many of us, Logo itself has served as a transitional object that has helped us comprehend a new way of using computers instructionally. Although we have not yet seen any journals devoted to Boxer, there are many journals and newsletters devoted to hypermedia applications. However, a careful examination reveals that the majority of these articles describe methods for production of CAI with these new systems. There should also be a place for exchanging ideas about use of these new systems for construction of the learner-based tools for which they are so admirably suited.

Over the years a number of people who were formerly active appear to have dropped out of the Logo community: Tim Riordon, Steve Tipps, Paula Cochran, etc. (We mention these names because we have coauthored Logo books with each of them, but we are sure that you could mention many others.) Rumors that these individuals have been abducted by the Logo police are completely untrue! In many cases, they are still actively using learner-based tools, but are employing a range of many different tools rather than Logo alone. We would like the *Logo Exchange* to follow their good example.

If there were any truth to the rumors of the Logo police, we would have certainly have heard from them by now as a result of writing this article. We can assure you that we will all be back next fall, writing our columns as we have in years past, although the content of the articles will continue to evolve. Er, one moment ... What's that?!!? They're coming in the front door! Glen, you go out the back. Gina, you take the manuscript. Run, Judi, RUN!!

References

Bull, G.L., Cochran, P.S., & Snell, M.E. (1988). Beyond CAI: Computers, language, and persons with mental retardation. *Topics in Language Disorders*, 8 (4), 55-76.

Bull, G. L., Lough, T. , & Cochran, P. (1987). Logo and exceptional individuals. In J. D. Lindsey (Ed.), *Computers and exceptional individuals* (pp. 169-187). Columbus, OH: Merrill.

Cochran, P. S., & Bull, G. L. (1985, November). *Creating a shared context: Using a computer in language therapy.* Paper presented at the annual convention of the American Speech-Language-Hearing Association, Washington, D.C.

diSessa, A.A., & Abelson, H. (1986). Boxer: A reconstructable computational medium. *Communications of the Association for Computing Machinery*, 29 (9), 859-868.

Gould, S.J. (1989). *Wonderful life: The Burgess shale and the nature of history.* New York: W.W. Norton & Company.

Mokros, J.R., & Russell, S.J. (1986). Learner-centered software: A survey of microcomputer use with special needs students. *Journal of Learning Disabilities*, 19 (3), 185-190.

Rosegrant, T. J., & Cooper, W. (1985). *Listen to Learn* [computer program]. Boca Raton: IBM.

Papert, S.P. (1980). *Mindstorms: Children, computers, and powerful ideas.* New York: Basic Books, Inc.

Paper, T.Q. (1980). *Gear Storms: Children, cogs, and powerful Machines.* Nuevo York: Alternate Universe Press.

Rosegrant, T. J., & Cooper, W. (1987). *Talking text writer* [computer program and teacher's manual]. New York: Scholastic.

Russell, S. J. (1986). But what are they learning? The dilemma of using microcomputers in special education. *Learning Disability Quarterly, 9*, 100-104.

About the Authors

Glen Bull has written a column for the *Logo Exchange* since it was founded. Gina Bull was originally a fine arts librarian, but changed careers as a result of exposure to Logo. She subsequently acquired a graduate degree in computer science and a position as a system administrator in a department of computer science. Glen and Gina Bull currently write a column, "Logo and Company," which addresses the emerging array of "Logo-like" tools. Judi Harris is a columnist for the *Logo Exchange*, and writes a Logo column for *The Computing Teacher* as well. Collectively the authors have written more than a hundred chapters, books, articles, and columns about Logo.

Editor's note: This manuscript was slipped over the transom and was lying on the floor when we arrived at the office one morning. Despite the fact that we have not yet been able to contact Bull, Bull, & Harris, we would like to reassure you that the conclusion to this article is a prank, resulting from what we can only describe as a distinctly odd (not to say sophomoric) sense of humor. However, if anyone should see any of these individuals, we would very much like for them to contact us.