Learning
in
Logo Microworlds

Editors: Anne McDougall
Carolyn Dowling

Computing in Education Group
of Victoria

Learning
in
Logo Microworlds

Editors: Anne McDougall
Carolyn Dowling

Contents

Learning in Logo Microworlds 1
Anne McDougall and Carolyn Dowling

Logo in Australia: 21 Years On 3
Jeff Richardson

Logo Learning: What the Learners Say 7
Pam Gibbons

The History of Mr Papert 20
Martin Boyle

Young Children as Programmers: Fantasy or Flight? 32
John Oakley and Anne McDougall

Teaching Programming Without Teaching Programming 52
Jenny Betts

MicroWorlds in a Year Seven Classroom 70
Narelle Best

Staring From Scratch 78
Craig Kerwin

M.L.C. Merging Microworlds, Minds and Multimedia 87
Steve Costa

Logo Supporting the P-12 Curriculum in a Technology Immersion School 92
Anne McDougall and Jenny Betts

Logo and MicroWorlds at Westall Primary School - A Four Year Tale 101
Tonia Chapman

Microworlds: Making the Connections between the Abstract Concepts in Primary Science 109
Craig Duncan

21 Years of Logo: Logo for the 21st Century 122
Nicola Yelland and Jennifer Masters

Stars and Sprites 133
Josie Hopkins

Learning in Logo Microworlds

Anne McDougall
Monash University
Clayton, Vic.

Carolyn Dowling
Australian Catholic University
Ascot Vale, Vic.

The papers in this collection were presented at the conference Learning in Logo Microworlds, held at the University of Melbourne in Parkville, Victoria in October 1996, under the auspices of the Australian Logo special interest group, OzLogo, and the Computing in Education Group of Victoria. This conference marked twenty-one years - the "coming of age" - of Logo work in Australia.

A companion volume to this one, Logo in Australia - Ten Years On, contains papers from the conference with that title held in 1985, and provides insights into earlier Australian Logo work. These first ten years saw, among other developments, the design and manufacture of the Tasman robot turtle, publication by Australian authors of several widely used Logo books, and the inclusion of Logo as one of the recommended programming languages for senior school Computer Science courses.

The papers in the current publication reflect more recent developments. As LogoWriter and MicroWorlds (as well as Geo Logo, Turtle Math and StarLogo) appeared here from development centres in the U.S.A., we have seen investigation and adoption of these environments in a variety of school settings.

A major development in this country during the period under consideration has been the influx of portable computers into many schools, frequently accompanied by across-curriculum use of LogoWriter or MicroWorlds. Associated with this has been

increasing use of thematic and project work approaches to curriculum planning, more collaborative, group and discussion work, and greater flexibility of timetabling. These trends are evidenced particularly in this collection in the papers by Tonia Chapman, Narelle Best, Anne McDougall & Jenny Betts, and Steve Costa.

As well as these cross-curricular applications, Logo use for enhancement of understanding of concepts in specific subject areas continues. Science applications of this type are described here in papers by Craig Duncan and Josie Hopkins, and Mathematics activities in one by Nicola Yelland & Jennifer Masters.

Student programming as a means for student understanding and expression of ideas about concepts and processes has always been valued in Logo settings. Student competence in programming is the focus for the paper by John Oakley & Anne McDougall, and issues in the teaching of programming are discussed in some depth in the paper by Jenny Betts.

Participants at the Learning in Logo Microworlds conference made it clear that there is great enthusiasm for and commitment to the ongoing development of educational computing environments based on the traditional "philosophies" of Logo but incorporating new developments in hardware and software. It is our hope that this collection of papers captures the enthusiasm and commitment so evident at the conference.

Logo in Australia: 21 Years On

Jeff Richardson
Gippsland Campus
Monash University

The history of Logo in Australia begins in 1975. Scott Brownell, a teacher from the island state of Tasmania, brought a magnetic tape copy of *Logo* from MIT to Hobart, to run on a PDP-11 at the Tasmanian Education Department's computer centre. He then recruited another Tasmanian teacher, Sandra Wills, and secured a rare and expensive robot turtle from the General Turtle Co. The ensuing project saw every school in Tasmania connected, with a teletype terminal, to the PDP-11. Sandra would load the turtle into the boot of her car and travel all over the island, moving from school to school. At each school, children would hook up the turtle to their terminal and use their remote Logo to control it. It's quite astonishing to think that some of the children are now in their 30s!

This work led to two technical breakthroughs in global Logo history. With the arrival of the Apple, Personal Computers came to rule the earth and distributed computing went into hiding for 15 years. Richard Miller, of the University of Wollongong in NSW, wrote THE first version of Logo to run on the Apple, specifically to drive the robot turtles in the Tasmanian project. In addition to collaborating with Richard, Sandra Wills had overseen the engineering of a small and relatively inexpensive floor turtle, the TassieTurtle. The TassieTurtle achieved a degree of accuracy and precision that had eluded similar r&d efforts in Edinburgh and elsewhere; and it could be run from a 5.25" floppy disk on an Apple. Once Terrapin and LCSI Apple Logos became commercially available, Tony Adams of RMIT in Melbourne, Victoria, was quick to provide low level procedures to enable both to control the TassieTurtle. When Seymour Papert visited Australia in 1981 he was moved to tears when he found himself

in a room surrounded by a swarm of buzzing, beeping robot turtles.

Seymour's visit built on the pioneering work of the Tasmanians to inspire a generation of Australian Logoists. Logo veterans still remember Seymour's dazzling demos using TI Sprite Logo, a version capable of degrees of parallelism and playfulness only recently reached once more in MicroWorlds and StarLogo.

The propagation of Logo in Australia was next boosted by the publication of two important and influential Logo books in 1983: Learning Logo, by Tony Adams, Anne McDougall and Pauline Adams, and Let's Talk Turtle, by Carolyn Dowling and Liddy Nevile. These two books opened up a range of activities and insights into the use of Logo, from the charming zoomorphism of the turtle to sophisticated computer science ideas. Tony Adams' implementation of Prolog (a very vogue language at the time) in just five pages of Logo code rivals the inspiring projects in Brian Harvey's books.

By 1985, Anne McDougall of Monash University, Melbourne, was able to convene a conference entitled 'Logo in Australia: Ten Years On'. This conference was attended by Logoists from all over the country. The diversity of the people in attendance, teachers, students, academics and parents, was matched by the panoply of Logo versions and platforms that were in use at this time: Commodore 64, Atari, Tandy, BBC and TI contended with LCSI and Terrapin's Apple Versions. Logo had become widespread and its enthusiasts had an energy and optimism that belied their limited resources. Everyone was working with not many machines and not much time. Although the educational establishment could not now ignore Logo, Logoists were still very much a guerilla element.

The next big leap forward for Logo in Australia did not come for another five years. During this time, LogoWriter and Lego TC Logo were previewed, then commercially released, to critical acclaim. Though successful, this success was limited, not by any limitation in these products themselves, on the contrary.

They were too good. Here was the Logo idea so plainly expressed, an all encompassing curriculum, 'pages', now ubiquitous, years ahead of their time, that to really use these Logos could only be done with an educational revolution. It was not forthcoming, and it seemed that Logo had reached a high water mark. Also during this time, Peter Carter of the University of Adelaide, South Australia, kept the Logo community nourished and entertained with his quarterly newsletter POALL and his brilliant book 'Thinking Logo'. Back issues of both are much sought after even today and are still well worth hunting down.

In 1989 a curious and uniquely Australian development began which was to be the strongest vector yet for the spread of Logo in Australia. Liddy Nevile, in conjunction with the Australian Council for Educational Research, began the Sunrise Project. The key element of this project was laptop computers. In the pilot schools where it began, Coombabah on the Gold Coast in South East Queensland and Methodist Ladies' College in Melbourne, entire cohorts of children between years 5 and 7 were given laptop computers. One per child. And each machine was loaded with LogoWriter, only. And the entire curriculum was conducted and expressed by the children as LogoWriter projects.

As anyone in the Logo community might guess, it couldn't NOT succeed. The children did everything, Geography, Ancient History, Biology, Music ... expressed as LogoWriter projects. The imagery of children with their personal laptops captured the imagination of both lay people and educationists nationwide. This popular success was almost inescapably technocentric, but this factor had an interesting twist. Schools which wanted to emulate the project (there have been scores and the number is growing rapidly) tended to swallow the thing whole. The laptops were the shiny baubles that attracted the interest, but as nobody really knows what to do with computers in education, LogoWriter, and now MicroWorlds, became the default software for laptop initiatives all over the country.

This narrative is not complete without mentioning two of the strangest and most challenging Logo books ever written, 'Turtle Confusion' and 'Turtles Speak Mathematics' by Barry Newell, director of the Mt Stromlo Astronomical Observatory in Canberra. They evoke Lewis Carroll as they present a Socratic dialogue with the Turtle. This turtle leads the reader through a series of puzzles and conundrums that show that even today, in the shadows of StarLogo, MicroWorlds and Lego TC Logo, there's still plenty of life in the plain old turtle.

Logo Learning: What the Learners Say

Pam Gibbons
School of Arts and Sciences
Australian Catholic University

Introduction

This paper uses transcribed student commentary to explore two central and recurring themes in my experience of teaching with Logo:

- being afforded an insight into the complexity and variety of the learning process;

  and

- seeing students become passionate about their own learning.

What my students have had to say about Logo has taught me a great deal about just how different the process of learning can be for different people. What is also quite remarkable is students' ability to analyse and articulate their own thinking when they talk about their Logo experiences. Beyond the learning theory, the insight into and expression of individual differences, the cognition and metacognition, the problem solving and critical thinking, there is however another aspect of learning with Logo that, as a teacher, I have found to be more rewarding than any other. That feature is its ability to inspire passion in learners about learning. From this student-centred perspective, this paper therefore endeavours to provide a focus for reflection on what might be termed the 'spirit' of Logo.

Why Are We Here?

In July last year at ACEC '95 in Perth, I came across a group of Logophiles who were feasting on a dessert of sticky cakes with almost as much enthusiasm as they were discussing the idea of having a conference to celebrate 21 years of Logo in Australia.

I remember the context clearly because, as people were introduced to each other, there was much anxious and unsuccessful hand-wiping going on before we plunged into the inevitable jam-and-cream-assisted sticky handshake.

While it may have seemed a bit awkward at the time, I look back at it now and think of what a perfect context it provided to start planning for a celebration of Logo's coming of age in this country. In my mind I began to think of the sticky handshake as something symbolic - here we all were, grown-ups, taking great delight in indulging in the joys of what others might dismiss as 'kid's stuff'. We were up to our elbows in it and we wanted to try it all. Once we had indulged, our meetings with others who had similarly indulged took on new significance - we knew we had shared a wonderful experience because, when we shook hands, we sort of stuck together. There it was decided that, just as in life, the best way to celebrate a transition into adulthood is to throw a big party - calling all of one's friends together to reminisce about the past and to look ahead to what the future may hold for the guest of honour - a wonderful child-like activity to celebrate growing up.

It was not until almost a year later, at ACEC '96 in Canberra, that I heard questions being raised about the wisdom of holding a 'Logo conference'. Suggestions that a conference focusing on a piece of software was 'technocentric' shocked me. I didn't think I was technocentric - hadn't I always been critical of technocentricity? And yet in my heart I believed that holding a 'Logo conference' was a great idea; more than that, I thought it would be fun! After all, I had bonded with people with sticky handshakes - how could something that felt so right be so wrong? And to have the words of Seymour Papert (1985) used against the concept was perhaps the most troublesome challenge of all.

I needed to resolve this contradiction before I could again feel comfortable about this conference. This became an even stronger private imperative when I was invited to be a speaker. I came

eventually to ask myself, 'Who is the real guest of honour at this party?'. I believe that the answer is not 'Logo'; I believe the real guest of honour here is 'Learning with Logo'. I am delighted that the conference organisers have, quite independently of my musings, reflected this subtle but crucial distinction in the name they have chosen for our party - 'Learning in Logo Microworlds'.

We don't have to look too hard to establish our precedent - a quick look back to the little book that started it all, "Mindstorms" (Papert 1980), is enough to establish that when we're talking about Logo, we're talking about learning. Logo may provide the stage and the actors, but the play is about learning. We know the feeling because we have been, and will go on being, Logo learners ourselves. Can you remember when you first used Logo? About twelve years ago, a friend put me in front of a Commodore 64 running Logo and said, "Type FORWARD 50 and see what happens." I don't think I had any expectations, I just did it. The little triangle rocketed up the screen and left a thin green line in its wake. I was amazed. "Wow, look at that! That's great! What else can it do?". My life was never really the same.

Three months later I was teaching Computing Studies. The school only had two pieces of software - a word processor and Logo. It didn't take me long to work out where all the fun was to be had; Logo provided a fantastic world in which my students could think critically, think creatively, solve problems, explore, understand the need for discipline and attention to detail, and still have fun. I couldn't get rid of them. Soon I was writing conference papers about the miracles I was seeing happening in my classes, and soon after that teachers from other schools started visiting my classroom. The latter surprised me greatly because, when all was said and done, I wasn't doing anything special - what it was really all about was what the students were doing with Logo.

I also remember my disappointment when, at the conclusion of conference presentations about the exhilaration of teaching and learning with Logo, people would rush to speak to me afterwards and start with questions like, "What version of Logo did you use?" or, "Why don't you use LogoWriter?" I was disappointed because I knew my presentation must have missed its mark. I knew that, for me, a particular version of Logo wasn't what I was excited about. Over the years I have used many different Logos - Commodore Logo, Amiga Logo, PC Logo, LogoWriter, Apple Logo II, PC Logo for Windows, Mac Logo, Object Logo for Macintosh - and Logo 'spin-offs' - The Phantom Fish Tank, StarLogo. I have used Logo with Primary, Secondary, and Tertiary students and all manner of other adult learners. When I look back at these various incarnations and think about what it is that is consistent and important from one to the next, I think of student-centred and student-controlled learning environments that are true to the Logo philosophy of empowering the learner. It is the wonderment of learning in a Logo microworld, and the ability to open that wonderment to others, that excites me. To teach with a tool that delights learners, that opens a window onto the elusive processes of learning for teachers and learners alike, that learners won't leave alone, that thoroughly engages learners, that breaks the bonds of indifference and incites surprisingly passionate love/hate relationships - these, I believe, are the embodiment of the 'spirit' of Logo that excites us all.

In this address I would therefore like to focus on the celebration of two gifts that a Logo microworld can give us:

- the privilege of seeing our students become passionate about learning; and
- the wonder of opening a window onto learning in all its richness and forms.

What Can We Learn From Logo Learners?

The irony of celebrating in the form of this address something which is so 'hands-on' and personally engaging is not lost on

me. If I've learnt anything from teaching with Logo, it is how much more effective and satisfying it is to have an audience that can be actively engaged in a process rather than just talking to them about it. And yet here I stand, with few options beyond talking to you. Perhaps the next best thing I can offer are the words of my students. Having collected over the years quite a lot of formal data to support my Logo-based research, I now have a wealth of transcribed commentary from learners collected before, during and after their Logo experiences (Gibbons 1989, 1993). In sharing their thoughts with you today, we might re-live, albeit vicariously, some aspects of learners' journeys through Logo microworlds.

What my students have had to say about Logo has taught me a great deal about just how different the process of learning can be for different people. What is also quite remarkable is students' ability to analyse and articulate their own thinking when they talk about their Logo experiences. I've seen the doers, and the dreamers, and just about every combination between the two. To begin with, even concepts of what the activity of programming with Logo is all about vary widely between students in one class.

When asked about the objective of programming with Logo, I received replies such as:

> *"Logo gets you to think about how things work. Most people just take for granted that things work but when you actually get down to a program, and you're actually going into the nitty-gritty bits of it, you appreciate it more."*
>
> *"I don't look forward to it. I know I've got to do it, so it's a job that has to be done."*
>
> *"It's challenging - it makes you think. ... I enjoy it because some of our other subjects are boring - it's not a matter of challenging to understand something; you have to go over something over and*

over and learn facts - I don't enjoy doing that, I like to use the brain to do more than just memorise; think through applications."

I also asked students to talk about how they approached a Logo problem. There were those who had no trouble analysing a problem and devising a sound strategy and yet could never quite make it work. Some had brilliant ideas and outrageous plans and derived a great deal of pleasure from working out problems 'from first principles'.

"I just look at the question and see what it asks and think of the best way to solve it."

"I should check up in books more, but I don't, I try and work it out myself logically, think it through. If I think I've heard it before and I know it, I don't like going back to it again - once I know it, I like to think I can remember things."

"I try to understand what the question is asking."

For some, visualisation was the key to analysing a problem:

"When starting the project I have a very clear picture of what the outcome will look like ... Clearly seeing the picture in mind, I then think of starting the project."

"I first think about what I want it to do on the screen ... I visualise it. Sit down and think about it."

There were those who struggled with designing an original solution and yet had a gift for recalling previous solutions and the fine detail of syntax:

"I try and find a procedure that I can compare it with for somewhere to start. Or I try to think back to one that I've done before as somewhere to start. If it doesn't work, then I try to modify it. So if I'm given a new problem, the first thing I do is look for a similar procedure, or try and think of something

that I've done that's similar, or just even one little part that's similar that could help in making it."

"You get all your data and notes that you've had in lectures and you see if you can work something out. You know, if I had something like this for an assignment or homework or something, I wouldn't sit down at my desk with just this - a piece of paper and a pencil - I'd get my notes out and hopefully there'd be something there."

And some had the gift of being able to discerningly combine design, structure and recall:

"Have we done anything else like it that can be modified or used? If there is I use it, if there isn't I just try and think what has got to be done and do it. ... I try and break it up into parts and see if I can solve a part at a time and then try and put them all together."

There were those who favoured elegant solutions and for whom the general concept and personal understanding were the crux of solutions:

"Well, you can see the pattern from this much." (To justify not needing to pursue a problem through to an obvious solution.)

"Coding? It's not really the important part of the problem."

"I think I'm different ... because I want to really understand it myself."

"I like to know what's going on. I like to understand exactly what's going on. I hate the idea of just learning a formula and then using that formula; I like to understand how that formula works."

"I don't want to do it if it's messy. I'm doing it this way but I don't like what I'm doing. Because it's

too complex, it's too ... I'm taking every little step - and there's got to be an easier way to do it."

And there were those who didn't mind how inefficient their solution was, as long as some solution was achieved:

"There'll always be easier ways, but that was the first approach that came into my head, and once you've got an approach where you think you can solve it, to me it's a bit difficult to try to get something else."

Students have plenty to say about implementing their solutions and the process of debugging. In what was clearly the stage of the problem-solving process which presented the greatest degree of frustration, students showed ingenuity in establishing techniques of working through the process in styles best suited to their own abilities:

"Sometimes it can only be one word wrong and I might try a whole new strategy."

"Sometimes it takes a lot of trial and error. Just testing out different things. Once I scrubbed the whole thing and just started from scratch. Because I didn't have the original in front of me so I wasn't trying to figure out what was wrong in the original program, I wasn't distracted by the old version. I started from scratch, put down what I thought, an overall plan - instead of just changing little bits. It helps, it's like just scrubbing your memory and starting from scratch, you have to rebuild it."

"It helps to know that STEP command too. You can really pinpoint ... you can tell exactly where the problems are. You can follow it right through."

"When I'm testing it and it doesn't work and I get really frustrated and think, 'What's going on?', so if I think something's not right, then I'll change that

bit and test it to see if that's what it is. If it's not, I put it back the way it was and try something else."

"You should retrace steps of what you've already put in. Are the steps I've previously put in, are they ... is what I'm doing now the correct next step from what I did before? If it's not, then I've got problems, so what can I put in its place?"

"I try and work out why it's not working - I look at each line of the program ... and see if it's just in the design of the program or in the typing."

"The first bit [pre-recursive call] is what you are telling the computer to do. The second bit [post-recursive call] is relying on the inputs that you put in but it has already used. So the first sort is okay because the instructions are still written down in front of you, but the second sort is hard because the computer remembers it, and it remembers it better than I can remember it. It's got a better memory than me. I think what I'm doing is jumping ahead and just generalising what I want it to do - instead of thinking about it and 'sussing out' what exactly I want it to do. Instead of taking it logically step by step, I'm taking a general leap."

What also surprised me was not just how willing students were to talk about their own attitudes and approaches to learning with Logo, but how astutely aware they were of their own approaches in relation to others in class - their seeing metacognition from another perspective:

"I know some of the others can sit down and they ... think about it for a while, like [Student A] and [Student B], they can do that, and then they say 'Oh, I've got it' and then type it in and it will work, or nearly work. But I have to sit with the computer from the word go and do a couple of lines, see what

> *that does, if it does what I thought it would, keep it and then add a few more. Mine's a lot slower. ... Like they've got huge plans before they come in and they sit down and go, they've got like 40 lines and I've got 1."*
>
> *"I look for similar procedures but [Student C] was using the knowledge that he's accrued without consulting anything else. Whereas I was looking through things."*
>
> *"Some can do it themselves and grasp it and go to work on it but others can't really do it by themselves. They look at someone else's answers and do it like that."*

If individual differences such as these can be identified, not just by the teacher, but by the learners themselves, how productive might be a problem-solving pairing of these individuals? Perhaps in a mediated Logo learning environment, informed by learners' own metacognitive awareness, we might move towards higher levels of collaborative learning and socially distributed knowledge. Logo microworlds might thus provide learning environments which, in the words of Rogers and Rutherford, "Allow individuals to use each other as resources and properties [and] ... develop a means of choreographing the flow of activity through individuals and technology." (1992, p.304).

Beyond the learning theory, the insight into and expression of individual differences, the cognition and metacognition, the problem solving and critical thinking, there is one aspect of learning with Logo that gives me more pleasure than any other. That feature is its ability to inspire passion in learners about learning. In this context, I am referring to the ability of learning in a Logo microworld to break the bonds of student indifference. It is rewarding to see students experience the mountains and the valleys and to see them realise that the lows enhance the highs, and that together, the experience is whole. Over the years

I have noticed that, with Logo learning, the agony and the ecstasy is a recurring theme:

> *"The feeling I felt after solving a complex problem was one of greatness, as if I had just accomplished an impossible feat. This feeling was similar to the feeling after we won our football final. What can I say ... Awesome."*
>
> *"If I do work on something, and I think about it, and it works - just the whole idea of it working, and you might say 'that's good' - it's satisfying. And if I find out that no-one else has done something before and I have - that gives me a buzz."*
>
> *"The frustration is part of the enjoyment of Logo - during and after. It's like when you've got a piece of a puzzle and you can't work out where it goes and you get frustrated and you can either put it aside or you suddenly find where it goes, and then the thrill!"*

So What Are We Celebrating?

Despite what I have seen in the classroom, I know that Logo struggles into adulthood. It battles to stay in the curriculum; it constantly faces the self-fulfilling prophecy of, "Oh, I heard that nobody uses Logo any more"; it fights the perception that 'real programmers don't program in Logo', that is, it's just for kids; in a discipline where obtuseness and mystique command respect, Logo is its own worst enemy - it's just too accessible. Perhaps we are true believers and, at the risk of mixing metaphors, I hope that I am preaching to the converted. So what is that we know that Logo can offer?

- It can offer a microworld of thought - I have yet to find a better context in which to explore de Bono's comment on metacognition: "Thinking about something is the only way to think about thinking, and having something to show for

thinking is the only way to judge its worth." (1967, p.13). On this point, my students also have something to say:

> *"In Logo you're not just repeating something, you're actually thinking. You're not just memorising you're actually using your brain to do stuff. And it can be applied to a whole range of problems - to everything, I guess."*

- It can offer a microworld which allows the articulation of ideas and provides a language to express ideas not just about what, but about how, we think. One of my most satisfying experiences as a teacher was when a student talked about his Logo learning extending beyond the classroom:

  > *"Do you know how we had that lecture when you explained it all, all this stuff? Well, I was driving home, and when I was driving around during the break ... I was thinking about Logo, and ... I thought about how it works and what you'd said, and I just worked it out to myself, so I kept repeating it over in my mind until I understood it. ... and I thought about it a couple of times, and I just thought how it worked. So when I came in I had a better understanding."*

- And it can offer a tool to help us rekindle the joy of learning in the learners in our care and gives us a window onto the process of learning - the diversity among individuals, the richness and complexity, the high and the lows - that is what I would like to celebrate today.

References

de Bono, E. (1967). The five-day course in thinking. Tiptree, Essex: The Anchor Press.

Gibbons, P. (1989). Confusion, cognition and metacognition: The agony and the ecstasy. In M. Dupé (Ed.), Backup the future: Proceedings of the 7th Australian Computers in Education Conference. pp. 123-126. Canberra: CEGACT.

Gibbons, P. (1993). Recursion: An analysis of individual differences in Logo programming. Unpublished doctoral thesis. University of Sydney.

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

Papert, S. (1985). Computer criticism versus technocentric thinking. Logo85: Theoretical Papers. Cambridge MA: Massachusetts Institute of Technology. Reprinted in Educational Researcher (1987) 7. 22-30.

Rogers, Y. and Rutherford, A. (1992). Future directions in mental model research. In Y. Rogers, A. Rutherford and P.A. Bibby (Eds), Models in the mind: Theory, perspective and application. pp. 289-314. London: Academic Press.

The History of Mr Papert

Martin Boyle
Monash University
Clayton, Victoria

Introduction

This paper outlines the life and career of Seymour A. Papert. We follow the development of Papert from his early formal and informal education in the South Africa of the 1930s through to his 'sage' era in *The Children's Machine* of the 1990s. On the way we will trace the formative influence of Papert's work with Jean Piaget in Geneva, his first serious collaborative work with Marvin Minsky, leading to the bombshell of *Perceptrons,* through to the flowering of his greatest educational achievement - the development of the computer programming language *Logo.* We will see how, in the later part of his academic career, Papert has drawn out the *constructivist* principles of Piaget into his own *constructionism* at work in the classroom. Anecdote spices the life of Papert with real humour and unexpected actions will give insightful glimpses into the workings of the great man's mind.

Early Life

Seymour A. Papert was born on March 1st 1928 in Pretoria, South Africa. The man himself seems shy to unwilling to divulge many details of his early life, though in *Mindstorms* he owns to an interest in gears from early childhood experiences, indeed:

> *Before I was two years old I had developed an intense involvement with automobiles.*
> (Papert 1980 p. vi)

Had the world then developed as it has now, such precocity may even have enabled him to enjoy a childhood sponsored by Henry Ford!

Papert further reveals a love for *Daisy* who

> *left me with an eleventh commandment: "Thou shalt invent three theories every day before breakfast and throw them away before dinner."*
> (Papert 1993 p. 58)

Such is the eclecticism of Papert's subsequent endeavours that one is inclined to accept that he fell, hook with the lot, for that beloved teacher's advice.

Papert's childhood gives every appearance of being outside the norm . His father was an entomologist who spent the best part of Seymour's early childhood roaming the east coast of southern Africa in pursuit of the tsetse fly; a life-style which required all members of the family to turn their hands in whatever direction was required and which, for a young boy, must have been more than just compensation for an eccentric upbringing.

Indeed:

> *The Papert family's way of life was straight out of a Hemingway story.*
> (Crevier 1993 p. 84)

The story goes off following bush trails, hunting for food and falling in love with the transmission differential of broken-down trucks. Strong formative experience for the young mind.

By the nature of things the Paperts were the only white people in the area and this led the young Seymour into trouble when eventually he had to attend school in Johannesburg. Unfamiliar with the strict and convoluted edicts of the political and social consequences of apartheid, the ten-year old organised evening lessons for the illiterate black domestic servants of his area and found himself in serious trouble with the authorities for such illegal activities; the consequences were to have interesting future reverberations:

> *This was just the first of Papert's anti-apartheid activities, activities that would later lead the*

> *United States immigration authorities to deny him a visa for many months.*
> (Crevier 1993 p. 84)

At the time the logic of the situation was lost on the young boy for:

> *Adults justified their reluctance to let blacks sit at school desks by citing fear of contagious disease. But, reflected Papert, these are the same servants who take care of babies and cook the food in the whites' homes. How can the ruling class think like that?*
> (Crevier 1993 p. 85)

Such interest in matters logical and illogical soon earned the talented boy an invitation to attend seminars in philosophy at the University of Johannesburg!

Early Career

Papert soon found himself a philosophy student at the University of Witwatersrand where he was awarded his BA in 1949. After this his interests switched to mathematics, leading to a Doctoral degree from Witwatersrand in 1952.

In order to widen his horizons Papert then moved overseas. He was awarded a Commonwealth research scholarship to St. John's College, Cambridge, UK, which would eventually enabled him to complete a second doctorate, and while in England he ran into one of America's foremost workers in the emerging field of Artificial Intelligence - Ed Feigenbaum - at the National Physical Laboratory outside London.

Feigenbaum was a Fulbright Fellow for the year, and

> *he had a memorable friendship with a young and somewhat eccentric South African scholar named Seymour Papert.*
> (McCorduck 1979)

More significantly for the future, at a symposium in London

itself, he first met Marvin Minsky. This was to prove the genesis of the second great collaborative effort of Papert's professional life, but first he turned his face towards France.

Papert spent the year 1956/57 as a researcher at the Henri Poincare Institute at the University of Paris, to complete the research for his doctorate, but then an opportunity opened and Papert was to spend a fascinating episode of his life as a researcher at the International Centre of Genetic Epistemology, at the University of Geneva, working under Jean Piaget.

> *... the Parisian discovery that had the biggest impact on my life was Jean Piaget, who at that time was giving a course at the Sorbonne. I got to know him and was invited to work in his centre in Geneva, where I spent the next four years and became passionately interested in children's thinking.*
> (Papert, 1993 p. 33)

Although Papert was to deviate from the 'gospel' of the master in terms of the rigidity of the stages of child cognitive development stages, the Pagietian influence pervades the remainder of his work fundamentally and overtly, contributing to such diverse concepts as the establishment and efficacy of computer-rich microworlds and the layering development of 'societies of mind'.

The Artificial Intelligence and Perceptron Saga

> *Following the four years in Geneva, I became a professor of mathematics at MIT. Many factors made the move attractive. There was the prospect of access to computers and of working with Marvin Minsky and Warren McCulloch, as well as a wonderful sense of playfulness that I had experienced there on brief visits. When I finally arrived, all this came together in all-night sessions around a PDP-1 computer that had been given to Minsky. It was pure play. We were finding*

> *out what could be done with a computer, and anything interesting was worthwhile. Nobody yet knew enough to decree that some things were more serious than others. We were like infants discovering the world.*
> (Papert 1993 p. 33)

One wonders how many of us have shared that heady experience with Papert when computers are new, computers are an unknown quantity - with computers anything goes!

In 1958 Marvin Minsky and John McCarthy had founded the Artif*icial Intelligence Group* at MIT. Minsky had not forgotten the young Papert who had made such an impression on him at their London meeting. After the previously mentioned visa problems were sorted out with the US immigration authorities Papert was in. He strode into Minsky's office, sat down, and they were away - never to look back!

> *Rarely had co-operation between two researchers been so productive: colleagues no longer said "Minsky-and-McCarthy," but "Minsky-and-Papert." The two soon initiated new research programmes in the theory of computation, robotics, human perception and child psychology. When the Artificial Intelligence Group formally became the MIT AI Laboratory in 1968, Minsky and Papert acted as co-directors.*
> (Crevier 1993 p. 86)

It was an exhilarating time in AI. Sane men and women with earned PhDs from highly respected seats of learning were claiming that the office thermostat was intelligent! Perhaps with hindsight and the passage of time they deserve to retain their now found anonymity!

Alan Newell and Herb Simon back in 1956 had produced software which could churn out proofs of theorems from Russell and Whitehead's Principia *Mathematica.*

> *In fact the Logical Theorist discovered a shorter and more satisfying proof to Theorem 2.85 than Whitehead and Russell had used. Simon wrote this news to Lord Russell, who responded with delight. However the Journal of Symbolic Logic declined to publish an article co-authored by the Logical Theorist describing this proof.*
> (McCorduck 1979 p. 142, footnote)

Minsky was building robots; Minsky and Papert were designing vision machines; and then there was chess ...

All day long the argument ebbed and flowed on the matter of intelligent machines. The philosopher Hubert Dreyfus published, in 1965, a report for RAND called Alchemy and AI (later expanded into the book *What Computers Can't Do*) generally panning what Dreyfus termed the 'artificial intelligentsia'. The ensuing battle was bloody and is, indeed, far from over!

Papert wrote the paper to refute Dreyfus, also for RAND, but the attorneys would not touch it!

> *It was eventually brought out as a Project MAC report with no lawsuits ensuing.*
> (McCorduck 1979 p. 196)

There's still great debate about who did what and who said what in the great chess debate. The fact is that a chess playing software package, which we will call Mac*Hack* throughout for simplicity, had been beaten by a ten-year old boy to the delight of Dreyfus, who was *alleged* to have claimed that chess playing computers could never beat any human player.

The program was strengthened (in fact it was a different program) and somehow Dreyfus, who was a poor chess player, was persuaded to play it.

Papert, smiling recalls,

> *"I organised the famous chess match. That was beautiful. He was - well, it wasn't all pathetic and*

> *sad because he was quite convincing. He was going to beat it very easily. And that also said something about him, almost naive. We didn't know. About halfway through we all thought Dreyfus was going to win."*
> (McCorduck 1979 p. 198)

Herb Simon had this to say:

> *Dreyfus thought that MacHack would play bad, mechanical, non-human chess. But it was a wonderful game - a cliff-hanger between two woodpushers with bursts of insights and fiendish plans ... great moments of drama and disaster that go on in such games.*
> (McCorduck 1979 p. 199)

Dreyfus was well beaten. Revenge for the 'artificial intelligentsia' was very sweet; but, as throughout history, successful houses soon turn upon themselves.

Frank Rosenblatt had been a classmate of Minsky at high school in the early forties. In 1962 he introduced to the press with, it must be said, some fanfare, the *perceptron.* The perceptron was a simple neural network, a model of artificial intelligence at odds with the then fashionable symbol manipulation models. Minsky had toyed with neural networks, in fact his PhD dissertation concerned them, and had dismissed their worth. Thus the claims made by Rosenblatt purporting to demonstrate the 'learning' powers of the perceptron were viewed right from the start with scepticism in the Papert camp.

The MIT faction did take the situation seriously though. David Waltz, a graduate student recalls:

> *Marvin and Seymour really were interested in Perceptrons. I and a bunch of other students took a seminar from them, where the goal was to figure out as much about Perceptrons as possible. We were merely to explore in a methodical sense what*

> *they were capable of and what they weren't, and try to characterize them in some way*
> (Crevier 1993 p. 107)

The upshot of all of this activity was the publication of a book by Papert and Minsky in 1969 which they called P*erceptrons*, and which demonstrated mathematically and very ably that the simple Perceptron was totally incapable of solving a wide range of important mathematical problems.

The repercussions of their book were immediate and dramatic, for Rosenblatt and his collaborators were totally unable to rebut the arguments. Neural network research was dead in the water amid claims of deliberate sabotage to divert federal funding away from networks and into symbol manipulation programmes. No self-respecting researcher would dare touch neural network research for a decade until the *connectionist movement* of the eighties which *has* proved greater potential for fruitful results. Connectionist researchers in AI blame Papert and Minsky to this day for the decade of neglect!

In the 1972 printing of *Perceptrons* there is a handwritten dedication to the memory of Frank Rosenblatt who was lost in a boating accident, by all accounts a broken man, shortly after the Perceptron affair.

We will leave the last word on the matter to Papert; he wrote in 1988:

> *Did Minsky and I try to kill connectionism, and how do we feel about its resurrection? Something more complex than a plea is needed. Yes there was some hostility in the energy behind the research reported in Perceptrons, and there is some degree of annoyance at the way the new movement has developed; part of our drive came, as we quite plainly acknowledged in our book, from the fact that funding and research energy were being dissipated on what still appears to me (since the*

> *story of new, powerful network mechanisms is seriously exaggerated) to be misleading attempts to use connectionist methods in practical applications. But most of the motivation for Perceptrons came from more fundamental concerns many of which cut cleanly across the division between networkers and programmers.*
> (Papert in Beakley (ed) 1992)

Mindstorms and Logo

We turn now to that most famous episode of Seymour Papert's life - the development of the computer programming language Logo. We will mention only very briefly matters of common knowledge. Suffice to say that the language had its beginnings in Papert's thoughts of the late sixties and lead to the publication of his seminal work *Mindstorms - Children, Computers and Powerful Ideas* in 1980.

The language was a development of the list processing language known as Lisp and its genesis from lists gave rise to its name - the Greek for *word*. The famous turtle probably finds its antecedents in a similar *living model* which used to roam the Bristol UK laboratory of brain physiologist Grey Walter in search of food by way of power outlets! The rest, as most would say, is history.

A second edition of Papert's book *Mindstorms* was published in 1993, and it is to the introduction of that edition to which we turn our attention.

As is right and fitting Papert uses the introduction to the second edition as a 'debugging' exercise for the first edition; practising his preaching on not getting things right the first time!

Papert acknowledges that he failed to anticipate the 'pick-up' rate of *Mindstorms* by teachers, especially elementary teachers, and is disappointed that many of his exemplary microworlds are classically physically and an impediment to such readers

going beyond Chapter 4 where the field is rich in Newtonian physics! Papert suggests that he would reorganise the work to present first the *Images of a Learning Society* to entice the elementary reader further into the plot.

Papert is also concerned with the notion, which he suggests some have read into the work, that computers will *cause* changes in the way children think. He replies:

> *What I was saying, and still say, is something slightly more subtle: I see Logo as a means that can, in principle, be used by educators to support the development of new ways of thinking and learning. However, Logo does not in itself produce good learning any more than paint produces good art.*
> (Papert 1993a p. xiv)

The emphasis in the first edition on structured programming is also regretted. Papert would much have preferred to introduce the tinkering bricoleur as an alternative programming style much earlier in the work, and he stresses his subsequent work in this field with his then wife the techno-psychologist Sherry Turkle.

The recent developments in Logo, in particular the MicroWorlds direction is seen as a move towards the encouragement of a richer and wider epistemological range and, we may say, a wider interpretation of what activities constitute computer programming.

Constructionism and the Children's Machine

In recent years the writings of Seymour Papert return ever more strongly to his Piagetian intellectual roots. In the late seventies his collaborative work with Marvin Minsky on the development of *Society of Mind* concepts shows, in its layering of different societies in the developing mind, the echo of Piaget's stages of readiness, and in the eighties his work with Sherry Turkle on epistemological pluralism and with Idit Harel on children as

software designers carries forward the *constructivist* notions of building knowledge structures into *constructionist* principles in the classroom.

Papert would consider a definition of *constructionism* as an oxymoronic concept but I suppose we will have to do with the banal, flat and constrained *learning by doing*.

Constructionism finds a true home in a computer rich culture and herein lies the heart of Papert's objection to current educational practice. He is not as might have appeared from *Mindstorms*, anti teacher. Rather he is anti the prevailing school culture which constrains children, physically and epistemologically, in pathways of its own liking.

In his closing address to the 1990 World Conference on Computers and Education he appealed, in the spirit of those times as an analogy with the political structure of the then USSR, for perestroika in epistemological politics.

As Papert says:

> *His [Mikhail Gorbachev's] slogan of perestroika (which literally means "restructuring") became synonymous with a policy of struggling to reform a system in serious crisis without calling into question the foundations on which it was built. It should be clear by now that I see most of those who talk loudly about "restructuring" in education in much the same light - though few of them have the courage to carry the reforms as far in their realm as Gorbachev didin his.*
> (Papert 1993 p. 206)

Perhaps the computer is The Children's Mach*ine* and the vehicle for freeing thought.

Endnote

We end with this vision of Papert:

> *Absent-minded like many driven intellectuals,*

> *Papert is said to have once realised, mid-way across the Atlantic, that he had left his wife behind in a New York airport. Colleagues report that he sometimes forgets to show up at lectures and, when he does, tends to get carried away into whatever topic fascinates him at the moment. A man of dramatic personal magnetism, he is likely to startle interviewers with juggling demonstrations at airport terminals or by stopping his car in the middle of a U-turn to formulate a thought. Papert's aphorisms, like Minsky's, tend to stick. One of his favourites is that we are to thinking as the Victorians were to sex.*
> (Crevier 1993 p. 86)

I will leave you to unwrap that saying of Papert's for yourselves!

References

Crevier, D (1993). *Artificial Intelligence.*New York: Basic Books.

McCorduck (1979). *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence.* San Francisco: W.H. Freeman & Company.

Papert, S (1980). *Mindstorms - Children, Computers and Powerful Ideas.* New York: Basic Books.

Papert, S (1988). The Conservation of Piaget: The Computer as Grist to the Constructivist Mill. In Forman, G & Pufall, P (Eds.) *Constructivism in the Computer Age.* Hillsdale NJ: Lawrence Erlbaum.

Papert, S (1992). One AI or Many?. In Beakley, B & Ludlow, P (Eds.) *The Philosophy of Mind.* Cambridge MA: The MIT Press.

Papert,S. (1993a). *The Children's Machine - Rethinking School in the Age of the Computer.* New York: Basic Books.

Papert, S. (1993b). *Mindstorms - Children, Computers and Powerful Ideas.* Second Edition. New York: Basic Books.

Young Children as Programmers: Fantasy or Flight?

John Oakley
Charles Sturt University
Bathurst N.S.W.

Anne McDougall
Monash University
Clayton Vic.

Introduction

The introduction of computers into education has been accompanied by the claim that programming of the computer by children can bring unique educational benefits. The research literature is inconclusive and the debate continues as to what constitutes "computer programming" and whether the claims made for programming are justified. The unique characteristics of programming are identified and questions raised as to whether the changing nature of the computer / user interface is leading to the demise of programming as an activity or simply changing the nature of programming. The purported values of programming as a problem solving activity and its contribution to cognitive development and transfer are outlined.

The propositions that programming is of little value for children and that programming's advocates' claims amount to fantasy are examined through several quantitative research studies. Some reasons for reports of failure to find the projected benefits of programming are suggested. These include the time frames used, the research questions investigated and research methodologies adopted as well as the educational context in which the programming activities took place. It is argued that many researchers failed to appreciate the nature of the claims made for programming or the nature of programming as a higher order cognitive activity.

In contrast to the quantitative research methodology used in the early studies there was developing concurrently a substantial body of evidence, both quantitative and qualitative, of benefits that were accruing to young children as a result of programming activity. Notable studies carried out in the USA were at Brookline in Massachusetts, the Lamplighter School in Texas and more recently, the Hennigan School in Boston; as well there has been interesting work in several Australian schools. The research methodologies adopted to assess the benefits of programming at these and other schools suggest very strongly that it is possible for young children to achieve "flight" as programmers and to experience cognitive growth and transfer.

There have been important changes in the nature of the research effort. The most important of these has been the move from quantitative to qualitative research with different types of questions being investigated. The research methodologies described in the more recent research literature generally address the criticisms levelled at earlier methodologies and their results are more in accord with the anecdotal evidence gathered during and since the Brookline studies. Sound research on children's programming indicates that for many aspects of cognitive development the claims for programming are justified. Increasingly the literature is identifying the importance of factors not addressed or even identified by early researchers as critical in any evaluation of the value of programming as a cognitive activity. Of these the study of the preconditions for "flight" is particularly important.

Programming Can Bring Unique Educational Benefits

Since microcomputers were first introduced into schools in the late 1970s there has been substantial emphasis on computer programming as an educational activity. The literature suggests that the underlying catalysts contributing to this belief have been the influence of those working in the field of artificial intelligence (e.g. Minsky 1968; Papert 1972) and the widespread

assimilation of constructivist epistemologies of learning after the work of Piaget (Papert 1980).

Essentially, the propositions arising from these catalysts were that students will learn about the process of problem solving by solving problems, and acquire knowledge through problem solving experiences. However, associated with those propositions is an additional suggestion derived from anecdotal evidence that the dual conditions of minimal available software and the availability of BASIC also provided a major stimulus for the introduction of programming in schools. Given this situation and a rapidly developing acceptance of constructivist views of learning, it is not surprising that programming has been viewed as a legitimate educational activity. The underlying supposition has always been that programming can bring unique educational benefits. This supposition was supported by more explicit claims for programming in general (Nickerson 1982; Pea & Kurland 1984b; Shiel 1981; Shneiderman 1980; Weinberg 1971), and for programming in Logo in particular (Harvey 1982; Kieren 1985; Lawler 1981; McDougall 1985; Papert 1980).

Nickerson (1982) cited by Pea & Kurland (1984a) summarises a widely held view of the benefits of programming:

> *Perhaps the basic reason for the belief that programming might be an effective vehicle for the acquisition of generally useful cognitive skills is the assumption that programming is prototypical of many cognitively demanding tasks. It is a creative endeavor requiring planning, precision in the use of language, the generation and testing of hypotheses, the ability to identify action sequences that will realise specified objectives, careful attention to detail, and a variety of other skills that seem to reflect what thinking is all about.*
> Pea & Kurland (1984a p.3)

While researchers such as Nickerson (1982) provided an extensive elaboration of the assertion that programming can

contribute to the development of cognitive skills, it was the claims of Papert (1980) that led to this becoming a more general belief. For many people, however, Papert was simply advocating the replacement of one programming language (BASIC) with another, Logo. This was not so and there was often a failure to recognise that "Logo the language" was accompanied by "Logo the culture" (Solomon 1986). The recognition of the importance of a culture that provides a unique environment in which programming takes place is a crucial element in any research on children as programmers. Unfortunately, following the publication of "Mindstorms: Children, Computers and Powerful Ideas" (Papert 1980)many authors made "exaggerated positive claims ... (that) appear so much more amenable to criticism than over zealous negative opinions" (Clements & Meredith 1993 p. 41).

The Research Literature is Inconclusive

To date the research literature relating to the benefits of programming as a tool for promoting cognitive development is inconclusive, and the debate continues as to what actually constitutes "computer programming" and whether the claims made for programming are justified.

Quantitative research has focussed on questions related to areas such as planning skills, the concept of recursion, mathematical skills, rule learning, cognitive style, social development, and problem solving ability.

If research is to be fruitful then it is essential that there exist some general agreement on what constitutes programming. Finding a widely accepted definition of programming is problematic (Kurland, Clement, Mawby & Pea 1987). Most authors would agree that programming has unique characteristics. Typically it has been defined in terms of activities such as "problem definition, design development and organization, code writing and debugging" (Kurland et al. 1987 p.104) or by way of a more general definition: "that set of activities involved in developing a reusable product consisting of a series of written

instructions that make a computer accomplish some task" (Pea & Kurland 1984b p.5). Increasingly, however, questions have been raised as to whether the changing nature of the computer / user interface is leading to changes in the nature of programming and therefore to changes in how programming is defined. It has been noted that over a period of four decades there have been major changes in those activities required to program a computer (Lockheed & Mandinach 1986; Pea & Kurland 1984b). David Smith (cited in Clements & Meredith 1993) argues that the traditional approach to programming is obsolete and suggests three "languageless" approaches that use direct manipulation of "pieces" as in HyperCard and MicroWorlds with their buttons, graphics tools etc; "demonstration" as in the creation of macros within a spreadsheet; and "visual rewrite" that in essence involves modification of an existing area of a screen. This raises the question of whether we are witnessing the demise of programming as an activity Lockheed & Mandinach 1986)or simply changing the nature of programming (Resnick 1993; Smith 1993; Kurland et al. 1987). Considerable debate has been generated by the proposition that the nature of programming has changed and that "languageless" programming has a legitimate place as programming in educational contexts (McDougall, Nicholson & Ferres 1995).

Regardless of whatever definition of programming is adopted or how that definition will influence research methodology, there exists also the need to consider the level of programming ability involved and / or required. The literature identifies at least four distinct levels: viz. program user, code generator, program generator, and software developer (Pea & Kurland 1984b). A program user (Level 1) requires no knowledge of programming and the user remains at the mercy of the program. Code generation (Level 2) requires a knowledge of syntax and the ability to write simple code. At the program generation level (Level 3) basic commands have been mastered and simple routines such as sorting, reading data, etc. can be written. The programs can be quite lengthy but lack refinement. Elementary debugging skills

have been acquired. At the software developer level (Level 4) programs are generated that are complex with sophisticated error traps. Code is optimised and software libraries and programming utilities are widely used. This level is characterised by careful planning, full documentation, and structure. Within each of these levels there are numerous sub levels. Pea and Kurland acknowledge that "there are many methodological problems with assessing programming abilities across these four major levels".

It has been argued that programming and working with computers can have a profound effect on individuals and their thinking (Papert 1985; Turkle 1984). Programming can be viewed by teachers as either a "treatment" with a planned and orderly sequence of instruction or a "cultural element" where the aim is to create a particular educational culture.

Any examination of the cognitive benefits that might flow from involvement in the programming process must consider the educational context in which programming is taught, the prevailing philosophy, the nature of the programming, traditional or otherwise, and the level of expertise reached by children as programmers. The consequences of these considerations are that researchers must specify questions about those elements and the associated cognitive activities and hence, presumably, the type of cognitive development likely to arise within those contexts.

Programming as a Problem Solving Activity

Associated with the concepts of generalizability and transfer is the widely held assumption concerning education in general, that the learning that takes place in the school context is generalisable and transferable. Lower order cognitive skills can easily be observed to be carried into adulthood. Most adults remember facts learnt as a child. Most retain skills such as being able to write, cut, paste, read and so on. However, it is less easy to identify the development of higher order cognitive skills and their transfer into adult type activities although most would

agree that such development and transfer do occur. Indeed, education is predicated on this assumption. Why have an extended period of schooling, often followed by tertiary education, if such transfer does not occur?

For at least two decades problem solving has been a focus for education and psychological research endeavours. Programming a computer has long been seen as one of those problem solving activities. Associated with programming has been the supposition that problem solving abilities and higher order cognitive skills developed as part of the programming process are also generalisable and transferable.

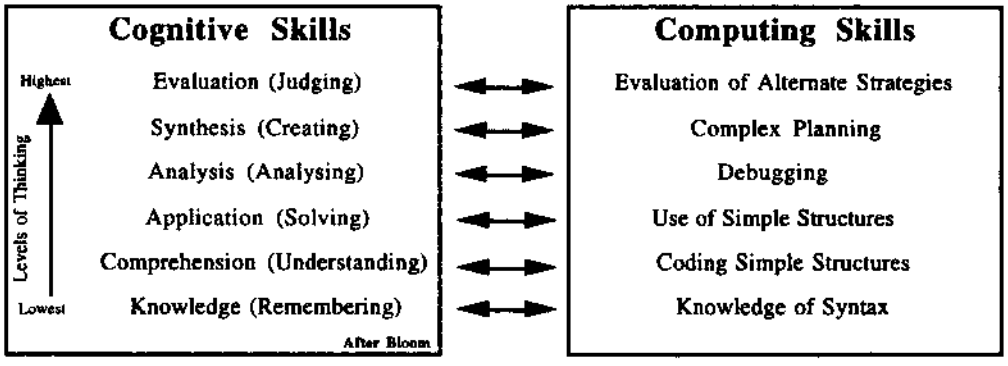
The difficulties in quantifying the development of higher order cognitive skills and their transfer are the crux of the dilemma facing researchers into the purported educational benefits of programming.

These benefits are claimed to be:

- the standard, by which we judge that programming is a worthwhile intellectual task, is that it has aspects which are "cognitively demanding." Specifically, the planning and debugging skills that are inherent in programming demand a cognitive involvement that goes beyond rote memorization or other low level thinking abilities. Additionally, planning and debugging are generalized skills which can be applied outside the domain of programming (Dalbey & Linn 1985 p. 254)
- task specification, sequencing, anticipation, hypothesis development, precision, consideration of alternatives (Nickerson 1982)
- planning (Feurzeig et al. (1981) cited by Pea, Kurland & Hawkins (1987)
- debugging - problem decomposition, procedural thinking - metacognition with opportunities (for children) to bring their thinking about thinking into the open as described by epistemologists (Papert 1980)

- debugging - estimates of 50 to 80% of programmer's time spent (Shneider, Weingart & Perlman 1978)
- reasoning skills needed for debugging - using feedback effectively, identifying patterns and inconsistencies, using deduction and inference (Weinberg (1971) cited by Dalbey & Linn (1985)
- the avowed purpose of most programming courses is to teach problem or reasoning as well as to teach programming.(Linn 1985 p.14)
- to promote thinking skills, problem solving, and related effects on students' cognition (Johanson 1988 p.3)
- as problem-solving experience accumulates, the systematic thinking will give way to an intuitive understanding of concepts and to the formulation of rules and strategies for solving problems (Singh 1992 p. 89).

These alleged benefits are perhaps best viewed in a context such as Bloom's hierarchy of cognitive skills, thus allowing us to categorise them as higher or lower order cognitive skills. The advantage of this categorisation is that it may facilitate decisions on what elements of programming are researchable using quantitative or qualitative methods.



The reality is, despite the rhetoric about developing higher order cognitive skills, that most teaching of programming in schools is atomistic and behaviourist in its approach with programming

consisting of the acquisition of a vocabulary of commands, syntactic rules, and the use of these in simple programming structures. This approach concentrates on the development of lower order cognitive skills. In contrast, few teachers teach programming in a manner conducive to the development of planning and problem solving skills and reflectiveness.

We would argue that elements of programming that require only knowledge, comprehension application and / or lower levels of analysis may be researchable using quantitative methods. However, most of the benefits of programming described above require higher levels of analysis, synthesis, or evaluation that are possibly best researched using qualitative methods.

Programming's Advocates' Claims Amount to Fantasy

Pea & Sheingold (1987) have drawn attention to the difficulties associated with researching the purported benefits of programming and alerted researchers to the significance of social and classroom organisational contexts and culture, together with teaching practices, as influences impinging upon cognitive development. They warn that any "changes ... appear to be slow to emerge and elusive to study". These observations seem to echo what was so eloquently stated by Papert (1985).

Given this warning it seems important to examine closely the propositions that programming is of little value for children and that programming's advocates' claims amount to fantasy. Most reviews of the literature in this field suggest that research has given us very mixed results. Many quantitative studies during the early 1980s suggested that the assertions of the proponents of programming had little basis and amounted to "technoromanticism" (Simon 1987 p.128; Pea & Kurland 1987 p.167). For example, the Bank Street College researchers (Pea et al. 1987) have made comments such as:

> *... unlike the knowledge of expert programmers. Programming constructs for the students had local functional meaning that they did not tend to generalize (p. 187)*

> *... most children appeared to do little preplanning ... Students tended to write and revise their code in terms of the immediate effects that commands and sequences of commands produced (p. 188)*
>
> *... students preferred to write programs interactively at the keyboard (p. 188)*
>
> *... we would argue that without some functional significance to the activities for those who are learning the new practices, there is unlikely to be successful, transferable learning (p. 195)*

Other negative findings are that: most students did not show evidence of transfer (Kurland, Pea, Clement & Mawby 1986); students who learn Logo fail to generalise this learning (Milojkovic 1983); and that there were no significant differences between Logo and non Logo groups (Horner & Maddux 1985).

By 1987 research on the impact of programming instruction on cognitive skills had yielded "occasional positive and many negative findings"(Saloman & Perkins 1987 p.149). Students generally programmed at a level considerably "below the ready fluency of the expert" with a concentration on mastery of skills (Saloman & Perkins 1987 p.159). The studies were generally characterised by short time spans and confounding differences in measures of transfer. Atomistic and behaviourist approaches to the teaching of programming are "not a very good way to foster cognitive skills"(Saloman & Perkins 1987 p. 164).

Some Reasons for Reports of Failure

Educationalists examining the results of quantitative research literature might well conclude that the claims of Papert (1980) and others are little better than "fantasy". Is this conclusion valid? Are there methodological factors that have contributed to the many non-supportive results? The answers are certainly worthy of consideration. Indeed, Pea & Kurland (1984a p. 44) comment, "if we think that learning general problem-solving skills is important, these results indicate that we cannot expect

this to happen spontaneously in the short space of a school year". The above dictum was ignored within their own research, e.g. "... over the course of the year ... children spent about 30 hours programming in Logo ..." (Pea et al. 1987 p.180). Furthermore, they stress the importance of other factors as confounding variables, e.g. "... it is by and large through interactions with skilled teachers that skill in problem solving and planning develops" (Pea & Kurland 1984a p.45). They identify additional inadequacies in some of their research designs, e.g. "Another objection to our planning tasks was that they were not close enough to programming tasks for the transfer of planning skills from the programming domain." (Pea et al. 1987 p.193).

Without doubt "... programming is a very complex skill, and therefore difficult to analyse and understand" (Dalbey & Linn 1985 p.268). In an attempt to reconcile the discontinuity between the predicted cognitive effects and the observed effects, Johanson (1988 p. 4) has proposed a series of hypotheses that includes a suggestion from Linn (1985) that "A cognitive chain of consequences exists: students are not progressing to the end of the chain, but could." This proposition is challenged by Kurland et al. (1986) on the grounds that students learn so little programming in the school context that it is not possible to progress to the end of the chain.

The hypothesis that the research on cognitive outcomes of programming has been poorly conceptualized (Johanson 1988) has been supported by Papert (1985) and surprisingly by Pea (1984). Of particular interest is the hypothesis that research has been unsophisticated and done at the wrong age level (Johanson 1988) which in effect recognises the possibility that there may be stages of Piagetian development that are more appropriate for the use of programming as a cognitive activity (Clements 1986). The use of Logo would provide the "raw" material for the development of this operational thought (Papert 1980).

However, the exposure to Logo also requires "a depth and

duration of exposure which probably has not been achieved in empirical studies to date" (Johanson 1988 p.7; Pea & Kurland 1987) with some exploratory studies involving only three hours of instruction (Rieber 1986; Webb 1985). Evidence is mounting that suggests that this hypothesis may be valid (Clements 1986; Degelman, Free, Scarlato, Blackburn & Golden 1986) and that a critical component of programming is metacognition (Bransford, Sherwood, Vye & Rieser 1986). In the discussion of an hypothesis related to "transfer" Johanson suggests that researchers may be approaching the problem incorrectly. This contention is supported by Mandinach (1987), Papert (1985) and Saloman & Perkins (1987). Papert's challenge to the traditional school curriculum (Papert 1980) is manifest in the hypothesis dealing with the discontinuity between problem solving, higher order thinking, divergent thinking and regular curriculum (Dalbey & Linn 1985; Hawkins & Sheingold 1986; Patterson & Smith 1986).

From these hypotheses and other evidence there emerges a list of crucial considerations to be addressed by researchers if the methodologies used are to have credibility. The reasons for reports of failure to find the projected benefits of programming can be summarised in terms of an inadequate understanding of the concept of programming as a "culture" versus programming as a "treatment", the limited time frames used, the research questions investigated and research methodologies adopted (Papert 1985)as well as the overall educational context in which the programming activities took place. There seems to have been a widespread lack of appreciation of the nature of the claims made for programming, the role of programming as a higher order cognitive activity and its sphere of cognitive influence. It would thus seem to be farcical to specify a single variable as the focus of a research question aimed at identification of higher order cognitive skills (which are notoriously hard to isolate or identify) when the teaching of programming usually occurred in a context that fostered only the development of lower order cognitive skills. Most quantitative studies attempted

to isolate a single factor in a complex situation. Few attempted to apply a broad spectrum of tests of cognitive activity, e.g.Clements & Gullo (1984). Papert (1985) argues that even these studies are flawed.

When reviewing the results of quantitative research into programming as an educational activity, we need to heed the warning that "research based on hypothesis testing ... can ... result in a series of experimental studies that only generate a vast amount of trivia" and that "to use classical experimental paradigms may be restricting efforts to investigate dynamic and flexible behavioral phenomenon" (Huck, Cormier & Bounds 1974 pp. 365, 372).

The Qualitative Approach

In contrast to the quantitative research methodology used in many of the early studies of the impact of programming, there was developing concurrently a substantial body of evidence, largely qualitative, of benefits that were accruing to young children as a result of programming activity. These studies were frequently described disparagingly as "anecdotal", a term that is akin to "ethnographic" or "qualitative". The implication was that such research is not "real" research, that if an effect can't be measured quantitatively, it doesn't exist.

Increasingly, research methodologists have come to recognise that ethnographic / qualitative research methodologies are often the best tools for researching complex phenomena. As Clements 1987 p.73) has noted, "its (Logo's) effects may not be immediate and direct, but delayed and diffusive". Papert and Minsky have encouraged researchers "to look for more qualitative outcomes"(Dalbey & Linn 1985 p.268) while Krendl and Lieberman comment that the relative merits of qualitative research are that it "focuses on what *does* happen in unstructured settings and provides broad description rather than isolating specific causal or mediating factors" (Krendl & Lieberman 1988 p.379). Only by adopting qualitative research methodology can the complex interactions of multitudinous variables be

identified. Notable examples of qualitative research on the effects of programming in schools exist in the literature. The first of these is the classic Brookline study (Watt 1979). Other studies worthy of mention include those of the Lamplighter School, the Canadian study of Carmichael, Burnett, Higginson, Moore & Pollard (1985), the Hennigan School (Harel 1986) and, within Australia, Ryan, Grimmett, Betts, Hallett & Mitchell 1991; Rowe 1993; John Paul College (Betts 1994), Elwood Primary School (Cole 1994); Shears 1995 and Batlow Technology School (McLean 1995).

The qualitative research methodologies used to assess the benefits of programming at these schools have identified a range of competencies and outcomes including a "change of culture" that have been developed. The teachers involved and external independent observers believe these to be a direct outcome of children being exposed to a particular type of computer programming environment. The nature and extent of these outcomes, just as with the merits of fine works of art, cannot be quantified but exist nevertheless. Within these schools young children have achieved "flight" as programmers and have been judged by experienced educationalists to have undergone, inter alia, cognitive growth and transfer.

Conclusion

The teaching of programming in schools has generally focussed on the lower order cognitive aspects such as coding. Due to limited allocation of time to programming activities, and other factors, students rarely have the opportunity to achieve "expert" status, and hence for the most part do not achieve levels of programming that utilise and develop higher order cognitive skills.

Despite the above, research questions addressed in quantitative studies, including those with positive results, have tended to focus on the development of higher order cognitive skills. It would seem to be an unreasonable expectation that development and transfer of these abilities would occur when students are

operating at levels of programming that require only lower order cognitive levels of operation.

There is growing consensus that the more complex the educational outcomes, the less appropriate the use of quantitative research methods and the more appropriate the use of qualitative studies. Qualitative studies attempt "to probe deeply and to analyse intensively the multifarious phenomena" and have the advantages of being "strong in reality", and pay "attention to ... subtlety and complexity" (Cohen & Manion 1989 pp.125, 150). We suggest that insights into the cognitive benefits of programming are likely to be best gained by the use of qualitative research methods in contexts where a culture involving programming has been developed. The emerging evidence suggests that the claims for programming will be vindicated and that programming will increasingly be recognised as an educational activity that can bring unique educational benefits.

References

Betts, J. (1994). *Becoming Immersed in a Technology Saturated, Logo Environment.* Paper presented at the Proceedings of the Australian Computer Education Conference, Brisbane.

Bransford, J., Sherwood, R., Vye, N. & Rieser, J. (1986). Teaching Thinking and Problem Solving. *American Psychologist, 41*(10), 1078-1089.

Carmichael, H. W., Burnett, J. D., Higginson, W. C., Moore, B. G. & Pollard, P. J. (1985). *Computers, Children and Classrooms: A Multisite Evaluation of the Creative Use of Microcomputers by Elementary School Children.* Ontario: Ministry of Education.

Clements, D. & Gullo, D. F. (1984). Effects of Computer Programming on Young Children's Cognition. *Gifted Education International, 2*(2), 129-133.

Clements, D. H. (1986). Effects of Logo and CAI Environments on Cognition and Creativity. *Journal of Educational Psychology, 78*(4), 309-18.

Clements, D. H. (1987). Longitudinal Study of the Effects of Logo Programming on Cognitive Abilities and Achievement. *Journal of Educational Computing Research, v3*(n1), p73-94.

Clements, D. H. & Meredith, J. S. (1993). Logo: Search and Research - Is Programming Obsolete? *Logo Exchange, 12*(1), 39-41.

Cohen, L. & Manion, L. (1989). *Research Methods in Education.* London: Routledge.

Cole, J. (1994). How I've Used Hypercard in Years Prep. to Six at Elwood Primary School During 1994, *Paper presented at the Annual Conference of the Computing in Education Group of Victoria*, . Melbourne: Computing in Education Group of Victoria.

Dalbey, J. & Linn, M. C. (1986). Cognitive Consequences of Programming: Augmentations to Basic Instruction. *Journal of Educational Computing Research, 2*(1), 75-93.

Dalbey, J. & Linn, M. C. (1985). The Demands and Requirements of Computer Programming: A Literature Review. *Journal of Educational Computing Research, 1*(3), 253-74.

Degelman, D., Free, J. U., Scarlato, M., Blackburn, J. M. & Golden, T. (1986). Concept Learning in Preschool Children: Effects of a Short-Term LOGO Experience. *Journal of Educational Computing Research, 2*(2), 199-205.

Harel, I. (1986). *Children as Software Designers.* Paper presented at the Logo'86 International Conference, Cambridge, Massachusetts.

Harvey, B. (1982). Why Logo? *Byte,* August, 163-193.

Hawkins, J. & Sheingold, K. (1986). The Beginning of a Story: Computers and the Organization of Learning in Classrooms. In J. A. Culbertson & L. L. Cunningham (Eds.) *Microcomputers and Education* (85th Yearbook of the NSSE, pp. 40-58). Chicago: University of Chicago Press.

Horner, C. M. & Maddux, C. D. (1985). The Effect of Logo on Attributions toward Success. *Computers in the Schools, 2*(2-3), 45-54.

Huck, S. H., Cormier, W. H. & Bounds, W. G. (1974). *Reading Statistics and Research*. New York: Harper and Row.

Johanson, R. P. (1988). Computers, Cognition and Curriculum: Retrospect and Prospect. *Journal of Educational Computing Research, 4*(1), 1-30.

Kieren, T. E. (1985). *Logo in education: What, how, where, why, and consequences*. Edmonton: Alberta Education.

Krendl, K. A. & Lieberman, D. A. (1988). Computers and Learning: A Review of Recent Research. *Journal of Educational Computing Research, v4*(n4), p367-89.

Kurland, D. M., Clement, C., Mawby, R. & Pea, R. D. (1987). Mapping the Cognitive Demands of Learning to Program. In R. D. Pea & K. Sheingold (Eds.) *Mirrors of Minds: Patterns of Experience in Educational Computing* . Norwood, N.J.: Ablex Publishing Corporation.

Kurland, D. M., Pea, R. D., Clement, C. & Mawby, R. (1986). A Study of the Development of Programming Ability and Thinking Skills in High School Students. *Journal, 2*(4), 429-58.

Lawler, R. W. (1981). *The progressive construction of the mind (Logo Memo No. 57)* . Cambridge, MA.: MIT Press.

Linn, M. C. (1985). The Cognitive Consequences of Programming Instruction in Classrooms. *Educational Researcher, 14*(5), 22-30.

Lockheed, M. & Mandinach, E. (1986). Trends in Educational Computing: Decreasing Interest and the Changing Focus of Instruction. *Educational Researcher, 15*(5), 21-26.

Mandinach, E. B. (1987). Clarifying the "A" in CAI for Learners of Different Abilities. *Journal of Educational Computing Research, 3*(1), 113-128.

McDougall, A. (1985). Teaching and learning about recursion. In A. Salvas (Ed.) *Communication and Change* . Melbourne: Computer Education Group of Victoria.

McDougall, A., Nicholson, P. & Ferres, G. (1995). Programming - Is It Dead in Your Classroom? In C. Dowling (Ed.) *Proceedings of the 1995 CEGV Annual Conference* . Melbourne: CEGV.

McLean, G. (1995). *Teacher Support Materials for the Application of Laptop Computers to the Primary Key Learning Areas*. Batlow: Department of School Education.

Milojkovic, J. D. (1983). *Children learning Computer Programming: Cognitive and Motivational Consequences.* Unpublished doctoral dissertation, Stanford University.

Nickerson, R. S. (1982). Computer Programming as a Vehicle for Teaching Thinking Skills. *Thinking: The Journal of Philosophy for Children, 4*(3), 42-48.

Papert, S. (1972). Teaching Children Thinking. *Programmed Learning and Educational Technology, 9*, 245-255.

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas*. Brighton, Sussex: Harvester Press.

Papert, S. (1985). *Computer Criticism vs Technocentric Thinking.* Paper presented at the Logo 85, Cambridge, MA.

Patterson, J. H. & Smith, M. S. (1986). The Role of Computers in Higher-order Thinking. In J. A. Culbertson & L. L. Cunningham (Eds.) *Microcomputers and Education* (Vol. 85th Yearbook of the NSSE, pp. 81-108). Chicago: University of Chicago Press.

Pea, R. D. (1984). *What Will It Take to Learn Thinking Skills through Computer programming?* Paper presented at the Annual Meeting of the American Educational Research Association, New Orleans.

Pea, R. D. & Kurland, D. M. (1984a). *Logo Programming and the Development of Planning Skills. Technical Report No. 16* . Chicago: Bank Street College of Education.

Pea, R. D. & Kurland, D. M. (1984b). *On the cognitive Prerequisites of Learning Computer Programming. Technical Report No. 18* . New York: Bank Street College of Education.

Pea, R. D. & Kurland, D. M. (1987). On the Cognitive Effects of Learning Computer Programming. In R. D. Pea & K. Sheingold (Eds.) *Mirrors of Minds: Patterns of Experience in Educational Computing* . Norwood, N.J.: Ablex Publishing Corporation.

Pea, R. D., Kurland, D. M. & Hawkins, J. (1987). Logo and the Development of Thinking Skills. In R. D. Pea & K. Sheingold (Eds.) *Mirrors of Minds: Patterns of Experiences in Educational Computing* . Norwood, New Jersey: Ablex Publishing Corporation.

Pea, R. D. & Sheingold, K. (Eds.). (1987). *Mirrors of Minds: Patterns of Experience in Educational Computing*. Norwood, N. J.: Ablex Publishing Corp.

Rieber, L. P. (1986). *The Effect of Logo on Young Children.* Paper presented at the Annual Convention of the Association for Educational Communications and Technology, Las Vegas, NV.

Rowe, H. A. H. (1993). *Learning With Personal Computers: Issues, Observations and Perspectives*. Hawthorn, Victoria: ACER.

Ryan, M., Grimmett, G., Betts, J., Hallett, K., & Mitchell, D. (1991). *The Queensland Sunrise Centre: A Report of the First Year*. Hawthorn, Victoria: ACER.

Saloman, G. & Perkins, D. N. (1987). Transfer of Cognitive Skills From Programming: When and How? *Journal of Educational Computing Research, 3*(2), 149169.

Shears, L. (Ed.). (1995). *Computers and Schools*. Camberwell, Victoria: ACER.

Shiel, B. A. (1981). The Psychological Study of Programming. *Computing Surveys, 13*(1), 101-120.

Shneider, G. M., Weingart, S. W. & Perlman, D. M. (1978). *An Introduction to Programming and Problem Solving with Pascal.* New York: John Wiley and Sons.

Shneiderman, B. (1980). *Software Psychology: Human Factors in Computer and Information Systems*. New York: Winthrop.

Simon, T. (1987). Claims for LOGO - What Should We Believe and Why? In J. Rutkowska & C. Crook (Eds.) *Computers, Cognition and Development* . Chichester: John Wiley and Sons Ltd.

Singh, J. K. (1992). Cognitive Effects of Programming in Logo: A Review of Literature and Synthesis of Strategies for Research. *Journal of Research in Computing in Education, 25*(1), 88-104.

Solomon, C. (1986). *Computer Environments for Children: A Reflection on Thoeries of Learning and Education*. Cambridge, Massachusetts: MIT Press.

Turkle, S. (1984). *The Second Self: Computers and the Human Spirit*. New York: Simon & Schuster, Inc.

Watt, D. (1979). *Final Report of the Brookline Logo Project.* Boston: M.I.T.

Webb, N. W. (1985). Cognitive Requirements of Learning Computer Programming in Group and Individual Setting. *AEDS Journal, 18*(3), 183-194.

Weinberg, G. M. (1971). *The Psychology of Computer Programming*. New York: Van Nostrand Reinhold Company.

Teaching Programming Without Teaching Programming

Jenny Betts
John Paul College
Daisy Hill, Brisbane

Introduction

Papert's vision of a new kind of learning environment in which there is free access between children and computers is becoming a more common trend today. In some cases the school provides each child with a personal notebook computer to call his or her own, and in other cases the child is asked to provide his or her own personal notebook computer. Either method supports the vision of Papert and his idea of students working in Mathland, a land where students learn to communicate naturally in a mathematical context. Either method supports the concept of immersion, a concept where a child can become engrossed in particular theme, or idea.

Observations made while working with students who are immersed in a Logo environment have confirmed that the more access a child has to a computer, the more adept the child becomes at applying Logo functions when exploring ideas, studying concepts or personally constructing knowledge. Students become more at ease with applying the Logo programming language in many different situations. Logo becomes a common language, a natural language. The more it is used, the more natural it becomes. Spending substantial amounts of time working with the Logo language has been made possible by the inclusion of personal notebook computers into the classroom. No more do students, nor teachers, have to book a lab or stand in line for a "turn at the computer". Access is now possible when and where it is needed.

Over time and with the right guidance, the user develops more skilful ways in which to use the Logo language.

Development of Immersion

My purpose is to provide the opportunity for students to naturally develop the Logo language that will enable them to take charge of the direction in which their knowledge could expand. There are three methods which I use to ensure a natural and individual development. Firstly, throughout most of the day students are encouraged to drive their learning towards a personal interest; secondly, those personal interests are used as a catapult for the development of skills and processes; and thirdly, I immerse the students in the Logo environment by using Logo in as many situations as possible during the exploration and study of compulsory curriculum components found in the Department of Education source books and current curriculum documents.

1. Launching a Learner-driven Learning Environment

Working with a common theme for all students simply provides a springboard for ideas as well as a common entity to which both the students and I can refer. Choosing a theme which provides enough scope to enable students to choose virtually any topic of personal interest, however, can become challenging. For example, themes based on "The Movies" or "Theme Parks" provide a myriad of topics in which students can explore the depths of a personal interest. After all, one could make a movie on any topic; one could base the construction of a theme park on any theme. It is during this deep-study that students apply their programming knowledge and will therefore explore the topic according to their own level of alibility. The more adept they are with the programming language, the more in-depth their work becomes.

Giving students the liberty to drive the direction in which their learning takes place also empowers them to choose a more natural learning path. However, it would be unprofessional and downright improper to leave a student alone to learn all that is

needed to be learned; therefore, teacher intervention is a critical component of this learning environment. When and how a teacher intervenes is determined by the direction in which the students move.

There are four phases in the development of a learner-driven environment.

Phase 1: The Plan - The Development

Students are responsible for choosing the topic and any sub-topics that are relevant. This is either done individually or as a group. The principal focus for me, as the teacher, is to facilitate the manner in which students organise their information and the manner in which students present the information. Dictating to the students how this is to be done is not the intention. Giving students the skills which will enable them to make more informed decisions about appropriate organisation and appropriate presentation is the focus.

The initial planning is quite a huge task. Emphasis is placed on the development of a plan of attack; this is labelled "The Proposal". Both the topic and sub-topics are listed with suggestions for how each area will be presented, the resources that may be required, the responsibilities of each group member, sketches illustrating screen layouts etc.

Much time is spent helping students to plan their time so they work efficiently and effectively. The proposal is meant to be very flexible and is never adhered to strictly. It is simply a way for students to organise their ideas formally, and something for me to use as an insight into the way in which they are thinking.

It is altered each time a new idea is carried out or an old idea rejected. It is altered each time a suggestion has been made or an idea has been explored. It is altered each time there is a new discovery or when an old discovery is modified. Briefly, it is a detailed record of the events which unfold over the period of time the student works on the in-depth study.

Phase 2: The Action - Working the Proposal

Students begin to carry out the designated tasks, either individually or in small groups. All actions undertaken by the students, or by me, are recorded on the proposal. Tasks may include researching, writing the Logo code, scanning appropriate pictures, gathering information, viewing videos, sorting through magazines or attending skills lessons such as learning how to create time lines, learning how to interpret maps, learning how to set out a letter, learning how to write a required piece of code etc. When it is evident that a particular skill is needed, intervention is necessary so that the student can carry out the task efficiently and effectively. Intervention takes place on either an individual level, a small group level, or a whole class level. There are a myriad reasons why intervention will occur.

- To demonstrate a skill that is required to complete the task. For example, scanning an article to find key words or learning how to operate a particular piece of software efficiently.
- To encourage students to approach a task in a different manner. For example, using the idea of a time line to display historical data. The time line could be created using the electronic medium and Logo or the paper medium.
- To provide students with resources. For example, ensuring students have access an appropriate video which will provide relevant information.
- To expose students to other sub-topics which students may choose to explore.

Phase 3: The Assimilation Process - Reflecting on the Action which Takes Place

Students are asked to reflect upon actions taken during the study. Group discussions take place with any students who are using similar ideas or any students who are working together as one group. Modifying the proposal is crucial during this stage as it

is the proposal that indicates the processes a student, or a group of students has taken throughout the study. Suggestions offered, problems encountered, solutions found, diagrams drawn, notes made, lessons attended and even lessons which were taught by a student are all recorded on this proposal by either the students or myself.

Comments made by me are noted on the proposal, alongside any comments the students make. The overall evaluation mostly involves the review of the processes used by students, with the end product playing a more minor role.

In many cases, students will move between phase 2 and phase 3 a number of times throughout the day.

Phase 4: The Outcome - Final Product

The proposal and the final manner in which the topic and sub-topics are presented form the outcome. It is not always necessary to wait until the very end of the entire study before presentations are made. Students who decide to present a small section of their overall outcome to the class before the in-depth study is complete are encouraged. An example of this would be the presentation of a play.

2. Reacting in a Learner-driven Learning Environment

Reactive lessons which come about because of personal interests of students generally occur on a day to day basis. Nevertheless, there are many times where one can predict the needs of the students. These lessons encompass a wide range of skills and process. However, for the purpose of this conference I shall refer to the development of learning to program with the Logo.

Although each student has individual needs as far as content is concerned, when it comes to using Logo, content becomes irrelevant. It becomes irrelevant because no matter what we are trying to explore, we begin to speak the same words, the same language, Logo. Even though students will vary in the way their work is presented, the Logo language still remains a common language. Students "talk turtle", regardless of whether they are

creating a screen with a map of Europe, a screen with a racing track, or a screen containing a beach scene. The task of getting the turtle from one place to another is one of the common links in many of the in-depth studies. Finding the common links is the key to knowing which reactive lessons to supply.

Reactive lessons take place on a number of different levels as not all students will need to participate. Students attend the lessons which are relevant to their current study. Lessons are categorised.

Classification of Reactive Lessons		
Nature of the lesson?		
compulsory	optional	invitational
Who is the teacher?		
teacher	student	
Type of lesson?		
individual	small group	whole class

This is a common pedagogy in many classrooms. However, the key element is the nature of the lesson; compulsory, optional and invitational.

1. Compulsory

These lessons come about for two reasons:

a) Reaction to the in-depth study. These lessons are whole class lessons and are initiated by me. However, they are not always taught by me.

b) Department of Education source books and curriculum documents specifying components which all students should cover. These lessons are either whole class lessons or small group lessons and are initiated by me.

2. Invitational

These lessons come about as a reaction to the in-depth study and are usually initiated by me. They are classified as invitational

lessons simply because not all students will be required to attend, therefore they are either small group lessons or individual lessons. If students are observed to be disorientated, I will intervene to initiate, organise and, depending on the skill being taught, arrange for the most appropriate person to take the lesson. My role as the facilitator is to is to invite students to participate in those lessons when I feel their current study may benefit.

3. Optional

These lessons come about as a reaction to the in-depth study and are the most common lesson. They are either small group lessons or individual lessons. Optional lessons are generally initiated by the student and are mostly taught by the students. At times, I will teach an optional lesson. At times, I even attend an optional lesson as a student.

All students are given the opportunity to participate in any of these lessons regardless of whether the idea is needed for their current in-depth study. If a child wants to participate "just because", then they are welcome to do so. Examples of optional lessons would be to gain an understanding of how to construct a special effect using Logo, to keep score during a game, to build a stopwatch, to design a better user-interface, to create an equation which will automatically calculate the amount of money being spent etc.

Devoting large chunks of time to the optional activities is vital as it allows students the time needed to analyse and reflect upon what they are trying to achieve. I do not feel compelled to have students complete these lessons in forty minutes or less. If the whole day is needed, then the whole day can be used to work through the problem. If a student chooses to spend the next three days working on the idea, then so be it. The combination of introducing notebook computers into the classroom and into the homes, accompanied by the students taking ownership of that technology, has now provided the opportunity for educators to immerse the younger generation in the medium and exploit the power this medium can provide.

3. Immersion During Learner-driven Learning

Building Experiences

In any learning environment it is reasonable to assume that the more time spent on building experiences, the more experiences will develop; therefore the more experiences that can be developed, the more experiences can be built. I like to think that the building of experiences is like a snowball rolling down a hill. The longer the snowball spends rolling down the hill, the more snow it gathers. If it can be directed to an area where the snow is plentiful then it will gather far more snow than if it is rolled in a snowfield where snow is scattered. Working in a Logo environment is similar. The longer a student spends working in an environment where the Logo is plentiful, the more experiences the student will gather. Perhaps this effect could be better expressed in a more familiar form.

```
to experiences :experiences
  pr :experiences
  experiences :experiences + experiences
end
```

My purpose is to provide the environment in which Logo is plentiful, and with the introduction of notebook computers into schools, Papert's vision is a reality.

> *... a new kind of learning environment [which] demands free contact between children and computer. This could happen because the child's family buys one or a child's friends have one...[or] let us assume that it happens because schools give every one of their students his or her own powerful personal computer.*
> (Papert 1980, p. 16)

In an environment where each student is expected to have possession of a personal notebook computer to use as his or her own, Papert's (1993) vision of students working in Mathland, a land where students learn to communicate

naturally in a mathematical context is very real. Students become immersed in maths and begin to use maths rather than do maths.

> *Geometry is not there for being learned. It is there for being used.*
> (Papert 1993, p. 17)

My focus is on students *using* the mathematical language of Logo rather than learning a scientific programming language.

Teaching Programming without Teaching Programming

At this point in time, each teacher who is in a classroom is obliged to include current sections of work from a Department of Education Source book or document, regardless of personal opinions. For me, it provides a grand opportunity to explore as many of those concepts with Logo. It is via those activities that the programming skills are imparted. The nature of these activities change over a period of time and are influenced by the skill level of students. As the months pass by, and as the years pass by, the essence of those activities changes dramatically. Activities fall into three categories.

1. Learning from a Model

For this type of activity lesson, the programming code acts as a model. It is recorded on a file. Students take a copy and simply use the code to explore the concept. Students are not expected to change or modify the code. The type of code used for this activity would be categorised as "advanced" for most students. What is advanced Logo?

> *Advanced Logo is what you don't yet understand*
> (Fitch 1994)

This is meant to be the case so that students are exposed to primitives, commands, styles and special effects that may be unknown to them. Experiences from these activities are invaluable when students are asked to construct their own solutions to problems (see *Learning through Constructionism*).

An example of this type of activity is the exploration of symmetry (see Appendix 1). Symmetry is part of the Space strand in the Queensland Mathematics Syllabus.

This activity models some important aspects of writing code:

- the superprocedure
- the subprocedures
- structured planning style
- simple tail-end recursion
- angles

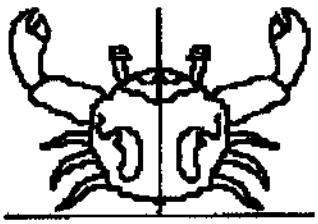


Figure 1. During the exploration of symmetry this design was created by a 10 year old student using LogoWriter.

This activity demonstrates the use of a number of common primitives and commands:

- t1 talking to particular turtle
- everyone talking to all turtles
- setc setting the colour of the turtle
- pd enabling the turtle to draw
- pu moving the turtle from one place to another without it drawing
- setpos placing a turtle in a particular position on the screen
- seth setting the heading of the turtle to face in a particular direction
- fd moving the turtle in a forward direction
- make creating a variable
- ascii using the ascii code to represent a particular character from the keyboard, in this case the four arrow keys

- if — running a list of instruction only if the first condition of an input reports true.

This activity also provides the user with what has become known as a "special effect".

The subprocedure **draw** provides the user with a model that will enable a user to navigate a turtle around the screen by pressing the arrow keys.

2. Learning by Debugging

For this type of activity lesson, the programming code still acts as a model. However, students are expected to key in the code rather than simply copy the file. Learning takes place through not only debugging the code as they key it into the computer, but when they are asked to modify the code to further explore the concept. Once again, the type of code used for this activity would be categorised as "advanced" for most students, and it is meant to be, so that students are exposed to primitives, commands, styles and special effects that may be unknown to them. Similarly, experiences from these activities are invaluable when students are asked to construct their own solutions to problems (see *Learning through Constructionism*).

An example of this activity is the exploration of sentence structure to develop the ability to control textual features (see Appendix 2), the grammar section of the third underlying element, skills, in the Queensland English Syllabus. Another example is the exploration of length by converting various measurement units (see Appendix 2). Length is part of the Measurement strand in the Queensland Mathematics Syllabus.

The concept of recursion can be used and modelled in many of these activities. For example, the opportunity arose in a Year 5 class when exploring numbers to 10 000 (see Appendix 2). Exploring whole numbers to 10 000 is part of the Number strand in the year 5 Queensland Mathematics Syllabus.

3. Learning through Constructionism

This is the simplest activity of all to implement as the teacher or student merely poses the problem, and solutions are derived from past experiences. It is this activity to which we aspire. The in-depth studies are a classic example of learning through Constructionism and accord with Papert's (1991) definition of a constructionist learning environment.

> *... a learning environment in which rich learning will come about in activities driven by enterprise and initiative.*
> (Papert 1991, p. 22)

The larger the bank of experiences, the larger the range of Logo code from which to draw ideas, and the larger the number of projects that can be explored. There are a myriad of activities. As the days pass, as the weeks pass, as the months pass, and as the year pass, learning through Constructionism takes a stronger foothold.

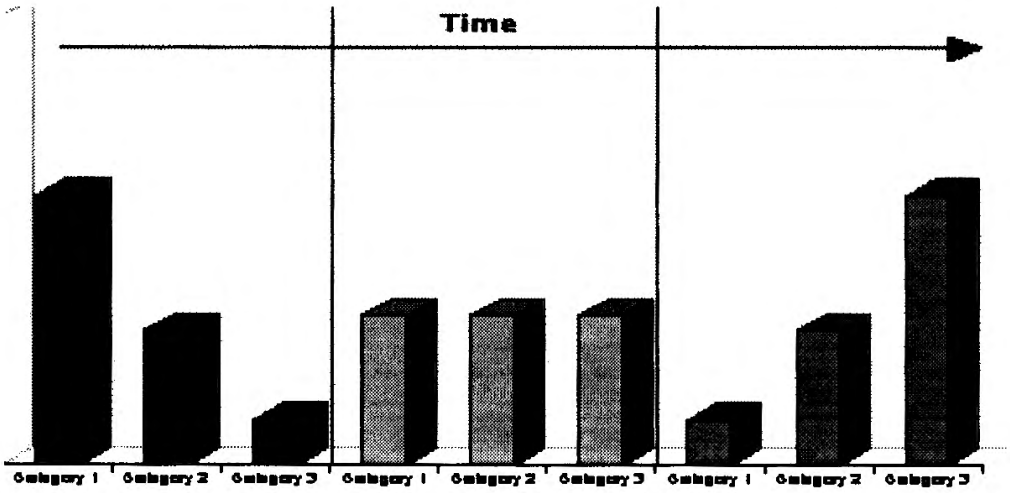


Figure 2. Implementation of different teaching strategies over a period of time

A Means to an End

Students never encounter a computer science lesson in programming. They simply use the code as a means to explore concepts, build projects, design software (Harel 1991) and construct knowledge. The more the students are immersed in Logo, the more "advanced" the Logo can become. The more "advanced" the Logo becomes, the more powerful the ideas can become. The more powerful the ideas can become, the more "advanced" the Logo can become ...

Concluding Remarks

Complete immersion in Papert's Mathland, a land where students learn to communicate naturally in a mathematical context, is a reality, and it is happening in learning environments where free access between student and computer is an obligatory part of the educative process. A Logo saturated learning environment provides a very powerful, natural, mathematical, learning environment. Two key elements enhance this learning environment: firstly, how often one exploits the power it provides, and secondly, the teaching strategies used during the implementation.

Today, the younger generation live comfortably in their computerised world. To be able to exploit the power of their world is their birthright and the responsibility of ensuring this occurs rests with the older generation, a generation who may need to modify pre-computerised theories to meet the expectations of the younger generation.

References

Andserson-Freed, Susan. & Brown, Lisa. (1995). *The Well Tempered Turtle: An Introduction to Programming Using Logo..* USA: Harvard Associates.

Department of Education. (1994). *Year 1-7 Mathematics Syllabus Support Document.* Brisbane: Department of Education.

Department of Education. (1994). *English Syllabus for Year 1 to 10*. Brisbane: Department of Education.

Fitch, Dorothy. (1993). *101 Ideas For Logo*. USA : Terrapin Software Inc.

— . (Winter 1994). What is Advanced Logo. Logo Update: *The Logo Foundation Newsletter* 2, 1-6

Harel, Idit. (1991). *Children Designers: Interdisciplinary Constructions for Learning and Knowing Mathematics in a Computer-Rich School.*. USA: Ablex Publishing.

Papert, Seymour. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. U.S.A.: Basic Books.

—. (1991). Perestroika and Epistemological Politics in Constructionism. In Harel, H. and Papert, S (eds). (1991). *Constructionism*. USA: MIT Laboratory.

—. (1993). *The Children's Machine: Rethinking School in the Age of the Computer.* USA: Basic Books.

Watt, D. & Watt, M. (1992). *Logo Learning: Strategies for Assessing Content and Process*. USA: ISTE Publications.

Appendix 1

The black turtle (T1) is the main turtle

```
to symmetry
 setup
 x-axis
 start-position
 draw
end

to setup
 t1, setc "black
 t2, setc "red
end

to x-axis
 cg
 t1, pu setpos [0 0 ] seth 90 pd
 fd 800
end

to start-position
 t1, seth 0
 t2, seth 180
 everyone [pu setpos [0 0] pd]
end

to draw
 make "arrow ascii readchar
 if :arrow = 72 [t1, seth 0 t2, seth 180]
 if :arrow = 80 [t1, seth 180 t2, seth 0]
 if :arrow = 75 [everyone [seth 270]]
 if :arrow = 77 [everyone [seth 90]]
 everyone [fd 1]
end
```

```
to symmetry
 setup
 y-axis
 start-position
 draw
end

to setup
 t1, setc "black
 t2, setc "red
end

to y-axis
 cg
 t1, pu setpos [0 0 ] seth 0 pd
 fd 500
end

to start-position
 t1, seth 270
 t2, seth 90
 everyone [pu setpos [0 0] pd ]
end

to draw
 make "arrow ascii readchar
 if :arrow = 72 [everyone [seth 0]]
 if :arrow = 80 [everyone [seth 180]]
 if :arrow = 75 [t1, seth 270 t2, seth 90]
 if :arrow = 77 [t1, seth 90 t2, seth 270]
 everyone [fd 1]
end
```

Appendix 1 continued

```
to symmetry
 setup
 x-axis
 y-axis
 start-position
 draw
end

to setup
 t1, setc "black
 t2, setc "red
 t3, setc "green
 t4, setc "blue
end

to x-axis
 cg
 t1, pu setpos [0 0] seth 90 pd
 fd 800
end

to y-axis
 t1, pu setpos [0 0] seth 0 pd
 fd 500
end
```

```
to start-position
 t1, seth 315
 t2, seth 45
 t3, seth  135
 t4, seth  225
 everyone [pu setpos [0 0] pd ]
end

to draw
 make "arrow ascii readchar
 if :arrow =  72 [t1, seth 0 t2,
  seth 0 t3, seth 180 t4, seth180]
 if :arrow =  80 [t1, seth 180 t2,
  seth 180 t3, seth 0 t4, seth 0]
 if :arrow =  75 [t1, seth 270 t2,
  seth 90 t3, seth 270 t4, seth 90]
 if :arrow =  77 [t1, seth 90 t2,
  seth 270 t3, seth 90 t4, seth 270]
 everyone [fd 1]
end
```

Appendix 2

Sentence Structure - Grammar	Equations that convert
to write setup make-sentence end to setup make "proper-nouns [Paul Gail Roland Rachel Jacky Patrick] make "verbs [runs talks laughs jokes sings] make "adjectives [Tall Sweet Gentle] make "adverbs [quickly loudly politely] end to make-sentence make "subject (se pick :adjectives pick :proper-nouns) make "predicate (se pick :verbs pick :adverbs) pr se :subject (word :predicate ".) end	to m-to-cm question [How many metres?] name answer "metres make "centimetres :metres * 100 announce (se :metres [m =] :centimetres [cm.] end to cm-to-m question [How many centimetres?] name answer "centimetres make "metres :centimetres / 100 announce (se :centimetres [cm =] :metres [m.] end to l-to-ml question [How many litres?] name answer "litres make "millilitres :litres * 1000 announce (se :litres [L =] :millilitres [ml.] end

Appendix 2 continued

Exploration of Whole Number to 10 000 - Counting

```
to go
 setup
 show-sequence :num
 check?
end

to setup
 make "questions 0
 make "num random 10000
 make "steps 1 + random 10
 ct cc
end

to show-sequence :num
 if :questions > 10 [make "final-number :num stop]
 pr :num
 make "questions :questions + 1
 make "num :num + :steps
 show-sequence :num
end

to check?
 question [What is the next number?]
 make "answer answer
 if :answer = :final-number [pr [ ] pr      [Correct] stop]
 pr [ ]
 pr [se [The answer is ] :num)
 pr (se [This sequence was going up in steps of] :steps
end
```

MicroWorlds in a Year Seven Class

Narelle Best
John Paul College
Daisy Hill, Brisbane

Introduction

> *... in terms of their own cognitive development, the child breaks away from rigidity, concrete and literal thinking and becomes more fluent, flexible and better able to grasp formal knowledge and create complexities."*
> (Harel 1991)

The use of MicroWorlds across the curriculum of year 7 encourages the students to expand their knowledge of both content and process as outlined in the syllabus and, most importantly, using the media of their time - the computer and Logo. The projects outlined in this paper are integrated into a class theme (where possible) and are fully integrated into the current curriculum. Most integrating devices for the themes are drawn from either the science or social studies curriculum, and other curriculum areas are drawn into the project.

The Participants:

The students participating in the study belong to one Year Seven class, consisting of 11 females and 17 males (28 students altogether). The students are in varying stages of their Logo development. Eleven of the students are in their third year of using Logo, the first year using LogoWriter and the subsequent years using MicroWorlds; ten of the students are in their second year of using MicroWorlds, and the remaining students are in their first year of MicroWorlds. All students have access to a personal notebook computer 24 hours a day, seven days a week. Students also have access to a variety of MicroWorlds reference

materials as well as other technology based research aids, e.g. CD-ROM and the Internet.

The students were introduced to the projects during theme/ activities time. All projects were introduced to the students so that time was allowed for the brainstorming of ideas and the immediate start of the project. Most of the projects were introduced at the start of the theme or unit of work and completed at the end of the theme. Some activities, such as the mathematics journal were year long activity, with students updating the project as new items were introduced.

MicroWorlds Projects:

Some of the projects completed throughout the year included:

- Clock
- Mathematics journal
- Pie graphs
- Volume
- Interactive magazine/brochure
- Haiku Poems
- Government data base

Clock

As this was the first project to be introduced to the students in the year, it was structured so that all new and non-confident MicroWorlds user could see an approach to solve the given project. This structure, adapted from Hammond (1984), allows the students to approach the project in various phases. This was found to be a satisfactory method as it allowed the students to see "a place to start". The planning phases are as follows:

Identify the problem to be solved

During this phase the project and expectations were explained to the students. The students were then allowed time to

'brainstorm' about various ideas that could be incorporated into the project. This phase was designed to be a whole class activity where all students participate. Once this phase is complete, those students who were confident with MicroWorlds were able to attack the problem themselves worked independently

Analyse the separate steps involved

This stage of the project allows the students to write down all the name of the superprocedure and the subsequent subprocedures they might need to define. In a project such as this the students recognised names of procedures such as: face, numbers, hands, move-minute-hand, move-second-hand etc. as being important. From this phase those students who needed a small amount of prompting could now begin to work independently. There can be an extra phase here where the students write down the names of the procedures in order. This phase is implemented depending on the students and their MicroWorlds development.

Code the program/do a test run/debug

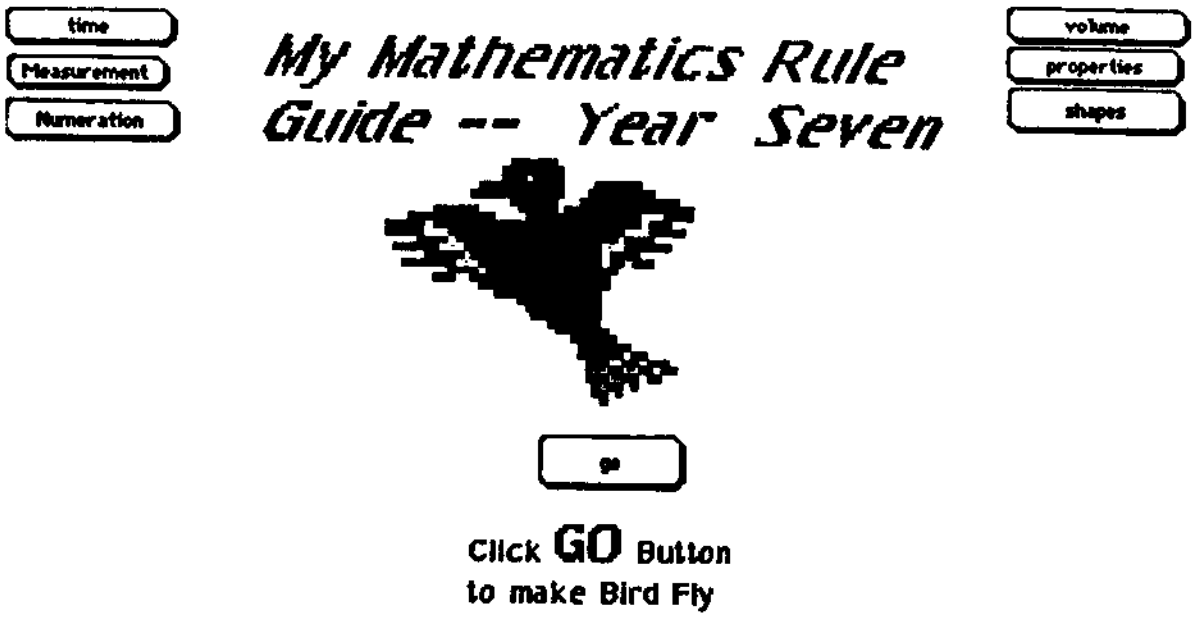
These phases were done simultaneously. The students were encouraged to key in the code to create the various subprocedures outlined in the previous phase. As they did this coding the students continually tested the procedures and debugged.

By using this approach it was found that the students were given a foundation from which to start. It is recognised that most students who were struggling were mainly having problems of 'where to start'.

Mathematics Journal

This journal was started at the beginning of the year and was added to throughout the year. The students set up a data base of terms that were relevant to their mathematics. Such overall mathematics headings as number and numeration, operations, measurement, space and chance and data were given to the

students (these heading are straight from the Mathematics Syllabus), and then other headings such as number systems, area, volume etc. were added to the database. How students moved into each of these areas was decided upon by the individual students. However, some suggestions included a menu system and buttons to various pages. Over the year as new terms, concepts and ideas were introduced to the students, they were required to add these to their mathematics journal.



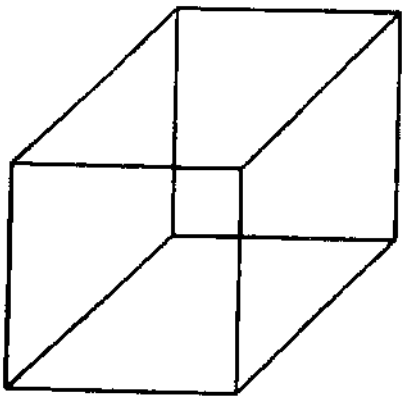
Volume

The main objective for this project was to reinforce the concepts already presented in volume. The students had experimented with volume using both concrete materials (water, buckets, sand) as well as working volume problems out in their mathematics book. This project required the students to program the drawing of three dimensional shapes using variables, then, from these figures to calculate the volume. The students were given, as a base, a copy of a procedure that would draw a cube (Kerwin 1994). The students then had to write procedures that would draw a rectangular prism, a cylinder and a triangular based

prism. All of the figures had to use variables to determine the lengths of the sides, the height of the triangle and the radius of the circle. The students then used these variables to calculate the volume of each figure.

During this project it was interesting to note the socialisation between the students. The students worked mainly in groups but there was an instance where two large groups were formed and a point of discussion was debated.

This project leads very well into discussion and exploration of perspective and proportion in art. Even though the diagram below is programmed to be a cube, they way in which it is drawn, makes it look more like a rectangular prism. The students, once they have drawn this figure, are encouraged to print out the figure and measure the sides to prove that it is a cube. This type of optical illusion leads very well into various concepts in art.



```
The volume of the prism is:

length x width x height

= 100 x 100 x 100 cm

= 100000 cm3
```

Pie Graphs

This project followed closely the phases as introduced above. The students found this project to be challenging,, thus most of the class followed the various phases. The students, during their studies on statistics and data gathering, had to design a program that allowed them to collect data about a certain question e.g. favourite colours of those in the class. From this data, the

program would then draw a pie graph showing the percentage of responses from the question. To make drawing the segments easier it was decided that the circle would have its origin in the 'home' position.

The students, using the primitive 'question', made a series of questions that related to what they wanted to survey. From the responses given, a pie or circle graph was drawn to match the percentages.

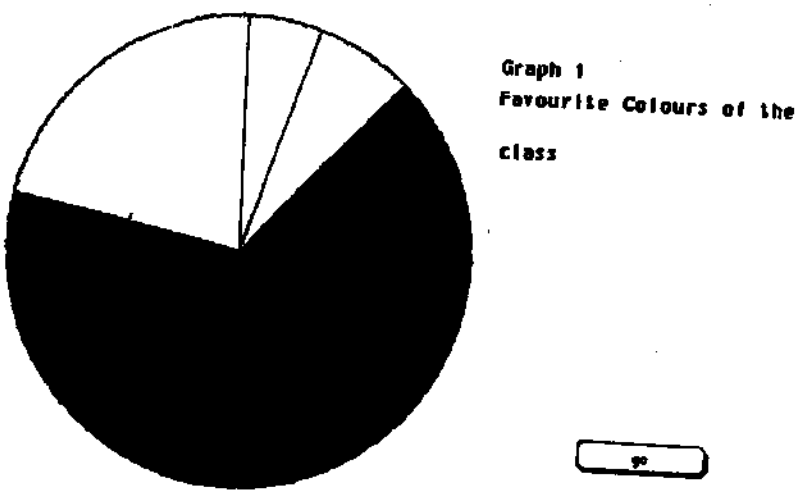
From this particular project the student outcomes were as follows:

- reinforcement of 360^0 in a circle
- collection of data
- interpreting and constructing visual representations of data collected.

It is interesting to note that the equation the students were working with to determine the number of degrees, i.e.

$$\frac{\text{number of responses}}{\text{total number of responses}} \times 360$$ is not formally introduced

until year eight mathematics. However, once this exercise was completed by the students the majority were able to understand how and why the equation was used.



Haiku Poems

As poetry is a genre that is appreciated, constructed and reinforced at all grade levels, to introduce the idea of *Logo* constructing the poetry interested all students. To introduce this concept and also the primitive 'pick' to the students, they completed a small program called "gossip". The students' names were placed into a category called "person", actions were placed in a category called "does-what". The computer picked from these variables to make a sentence.

After the students were used to the idea of the primitive 'pick', they were then asked to brainstorm about topics and phrases that could be put into a haiku poem. The structure, i.e. 5 syllables, 7 syllables, 5 syllables was emphasised to the students. The students then entered these phrases into subprocedures named '5-syllables' and '7-syllables'. From this, various haiku poems were devised. Once the program had been run a few times the students needed to manipulate the phrases so that the poems made sense.

The following projects were incorporated into the themes of the class:

Interactive Brochure - Journeys Around the World.

As the persuasive genre (debates and brochures) is a key genre for year seven to study, it was decided that the students would produce a brochure using the medium of MicroWorlds. Through exposure, the students became aware of the structure and type of language associated with brochures. In context with the theme, the students were asked to produce a brochure advertising a foreign country. The brochure had to be over three pages long with animation on each page. Other technology such as scanners was available to the students for the graphics. Using resources such as the CD-ROM, Internet, books and travel magazines, the students researched their chosen country and then made their brochures in MicroWorlds.

Government Data Base - Responsible Participation

In this activity the students had to build a data base that outlined the various structures of government. As the structures of the government were formally learnt by the students they added this to their data base. Again the students used a variety of ways to move from one page to the next. The students also needed to do a diagrammatic outline of the complete structure of the government. As some of the parts of government are not covered in the syllabus, e.g. the high court, the addition of these elements into the outline allowed for extension research.

Throughout the year, all students were exposed to a variety of projects integrated into numerous curriculum areas. While projects such as the government data base and haiku poems dealt with the processes of knowledge, direct content was incorporated through projects such as volume and the interactive brochure. Overall, the students were allowed to immerse themselves in MicroWorlds while learning the necessary curriculum.

References

Department of Education Queensland (1987). *Primary Mathematics Source Book.* Brisbane: Department of Education Press.

Department of Education Queensland (1988). *Primary Science Source Book..* Brisbane: Department of Education Press.

Hammond, R. (1984). *Forward 100..* Middlesex England: Penguin Books Ltd.

Harel, I. (1991). *Children Designers.* New York: Ablex Publishing Co.

Kerwin, C., (1994) *LogoWriter Turtle Solids.* Unpublished Worksheet. Brisbane.

Starting from Scratch

Craig Kerwin
The Glennie School
Toowoomba. Qld.

The intention of this paper is to share with you a story of how I attempted to create a Logo environment at our school.

There is no argument that Logo allows students to become creative learners. Through developing personal projects in a multitude of areas, both problem solving skills and creativity can be overtly promoted. As Dale Spender (1995 pp. 111-112) commented: "One has only to observe a class of students doing schoolwork with the same commitment, concentration, enthusiasm and sheer enjoyment that we normally associate with video games, to know that this (using Logo) is one of the most intellectually expanding and demanding of processes."

My experiences in a Logo environment commenced several years ago at John Paul College where many students and teachers were able to be completely immersed in computer technology, coupled with the daily use of LogoWriter and MicroWorlds in their learning environment.

Having quickly become exposed to LogoWriter and the Logo programming language during the implementation phase of their notebook computer program, and, sharing this with colleagues and students, I very easily became accustomed to working in a "constructivist environment".

The problem for me came when I accepted my new position at The Glennie School. While there was very little computer use, "constructivist environments" were few and far between. It didn't take long for me to ask the question: "How do I develop a Logo environment with students and teachers who have mostly

never seen or used MicroWorlds and where access to computers is currently limited?". Would I find myself divorced from MicroWorlds? Perhaps it would be separation? Or could this possibly the blossoming of a new romance with the turtle? I have to admit, with all that comes with a new job in a new location, that it was easier for MicroWorlds and me to decide to separate, even if only for a short period of time.

So why opt for turtle separation? As I mentioned, I came to this new school with a background in individualised computing using notebook computers, and while the same environment will eventuate at Glennie in 1997, the situation was completely different at the start.

While the computer technology in the school was given a major facelift, computer studies was introduced for the first time to students in Years 8 to 10. This involved using the two new computer labs to provide the ladies with two computer periods a week: one for word processing skills and one for computer awareness. Not the ideal situation, but at least a step towards more computer access. You must realise that the School had come from virtually nothing in the way of technology and that the development of two computer labs was only an interim measure as the school and its staff prepared for the introduction of notebook computers in 1997.

It is easy to see that "an education system which has traditionally functioned on the premise that anything that is fun cannot be proper learning will have difficulty adjusting to this new constructivist style of ebullient intellectual activity," says Spender. (1995 p112). It wasn't surprising that discussions held with teachers revealed that those with Logo experiences were few and far between and, of those, the experiences had not necessarily been pleasant. The chances of enthusing them, initially, seemed dim.

Most of the girls in Year 8 were very excited about the different opportunities they had had to learn about their computer. I knew

I couldn't change the school overnight, but at least I could make a start in the computer awareness classes and what better place to start than with the Year 8s. In discussions with the girls I learned that they would be interested in learning how to design a set of instructions the computer could follow - to design their own computer program.

This was very well received and much curiosity was expressed by the girls about how this would be possible.

My response to them was that we would use a program called "MicroWorlds". While at the time this meant nothing to them, they were eager to explore this new software package. The separation period was over.

There's something special about the first time someone new to MicroWorlds views the screen. "There's a turtle on my screen!" was the reaction of just about every girl and while many waited then for the turtle to do "something", others discovered that the mouse gave them the power to control where the turtle sat on the screen and the direction in which it was facing. The intuitiveness of the program was already hard at work.

I've always believed it is important for "turtle lovers" to learn to nurture their turtles using the right lingo: learning to "turtle talk".

It's important that first time users of Logo learn a little about the language from someone with experience and so there are several steps I follow.

The teacher acts as a turtle. The students provide the teacher with instructions. For example, "move forward please". The teacher moves forward until bumping into something in the way, like a table. Establish that the turtle is initially a little dumb and really needs things to be spelled out. Perhaps a distance would help. Students may respond with "move forward 40 steps please" to which the teacher responds.

Now have the students type in similar commands at the computer. When the turtle responds with "I don't know how to

move forward 40 steps please", it is time to establish that the turtle is also very lazy and likes it instructions to be kept short and simple. Could "move 40 work"? Try all options until "forward 40" is stumbled upon.

Once turtles are moving around, the teacher goes to the next level and explains that in turtle talk it is often possible to shorten primitives so that forward becomes fd.

Having established "fd" with the students, discovering "bk", "lt" and "rt" is easy. Used with "pd", "pu", "cg" and "cc", there are enough powerful primitives to allow the students to do some high level exploration without teacher intervention.

Once the girls could see the basic control they had of the turtle, it was time to have some real fun: starting with dressing the turtle in a new costume and then giving it a more exciting background to work with using the painting tools.

Gary Stager (cited in Spender 1995 p. 112) points out, "Five year olds and university professors experience the same playful enthusiasm towards problem solving and learning when working with MicroWorlds". The girls had had their first MicroWorlds' experience!

With this introduction, however, came a problem. The girls would only come in contact with MicroWorlds and the turtles for one 50 minute activity each week. I could try to teach them everything about the software from a programming perspective. I wasn't happy with this thought though.

Anyone who has read Papert's "The Children's Machine" has an understanding that this situation is not ideal.

I decided it was better to attempt to create a Logo way of thinking for the girls by incorporating the use of MicroWorlds with one of their other subject areas - the way in which it would be used if the girls had access to computers and MicroWorlds more often.

From discussion with the Head of Mathematics I decided to set the girls a maths project. They were to design a digital book

which could be purchased, theoretically, on disk from a book store. The book was to be a digital teacher. It was to teach a concept being covered in mathematics classes during that term. At that time the main focus was angles.

Using this method I would be able to encourage the girls to develop projects which made use of graphics, text, buttons, and, of course, Logo programming.

Before letting the girls attack the problem I took them through a few more essential steps. This included discussing with them what an object was, how to use the MicroWorlds objects, how to name these objects, and how to set instructions for objects.

Then it was time to start on the project.

It was important that I didn't have 26 girls in a class all produce the same project. The beauty of MicroWorlds is that it allows each person to use personalised programming and on-screen display. It was equally important that the girls did have direction. I have always used the following model as a focus for developing a MicroWorlds project.

Step 1. Establish the Problem

Ensure that the person completing a project understands the requirements of the project and there are sufficient challenges to allow creative thinking to occur.

Step 2. Strategies for Solving the Problem

Think about how the problem can be solved. If necessary, draw some diagrams or scratch out some rough notes to help in the problem solving process. This will help in maintaining focus on the original problem rather than being sidetracked by other issues. For example, adding graphics to a page which doesn't yet solve the original problem.

Step 3. Is the Problem Solved?

Each time something works successfully in a project, check that the original requirements of the problem have been met. If these

requirements are not yet completed, move back through the second step.

Step 4 Extension to the Problem.

If the problem is solved to this point, are there additional challenges which would make the problem more interesting? This step allows students to be extended in their MicroWorlds knowledge through further exploration.

Let's review the problem solving steps involved with the digital maths book.

View sample activities constructed by students.

Activity 1	The turtle draws a right angle.
Activity 2	The turtle draws other set angles.
Activity 3	Can the turtle draw an angle defined by the user?
Activity 4	Let's make a teaching tool: design an angle quiz. The turtle draws an angle and the user guesses what sort it is.
Activity 5	Possible extension - keep a score.
Activity 6	Add a colourful menu.

We all realise that, in general, there is no "sell job" required to enthuse children to use computers: this is a notion left for the adults of the world. No matter what school one works in, there will also be the rock-solid types who will continue to believe that computers corrupt the minds of the users and that software like MicroWorlds is merely giving kids an excuse to waste time playing.

I am not alone however. The Glennie staff were keen to find out more about the turtle and what it had to offer. Teachers are busy, they don't want to waste time experimenting with software when they can't see the purpose of why they are using the package.

I decided at a recent staff inservice day to offer two approaches:

a session called "What is MicroWorlds" and another called "What can MicroWorlds do for me?". In the first session I discussed the constructivist philosophy with the teachers before basically allowing them to play, much in the way the students had done during the term. These teachers enjoyed this approach and many "discovered" aspects of the software without needing any guidance.

In the second session I had installed several sample projects on each computer. These were kids' projects, not the glossy sales projects installed in the PB directory. There were typos, buttons which didn't work, procedures which had errors, and graphics which weren't finished. All showed the true glory of the MicroWorlds package.

The inservice session went for one and a half hours and these teachers were encouraged to look at as many of the sample projects as they could in about half of that time. In the second half I stepped them through a simple project, using the step method, and resources from my three MicroWorlds books. This gave the teachers a feel for what a Logo environment would be like in their classroom. The importance of this was that the staff had now experienced a safe, user-friendly Logo environment in which "getting right" didn't actually matter.

I had no doubt that MicroWorlds would be well received in the school. It was more a case of how to do that without having constant access to the software.

To be honest, not every child could see its purpose, but then I would have been naive to expect every student to want to use the software without having daily access to the package. It was certainly pleasing, though, to see the girls continue to make use of MicroWorlds, especially when they were given "free time" to continue using whatever package they wanted.

More staff, on the other hand, could see the relevance of the software, especially after having the opportunity to view projects previously worked on by students.

Since the inservice day numerous staff have made the effort to ask me for assistance in preparing to use MicroWorlds in their subject areas. That's definitely a plus!

Whether you are in the category of "starting from scratch" or you've been using a Logo programming package for some time, you might like to take the time to visit these Internet Home Pages and Resource Sites. You'll find many useful resources and links to other places in your travels here.

- Jenny Betts http://www.powerup.com.au/~jbetts/index.html
- David Potter http://iccu6.ipswich.gil.com.au/~dpotter/index.htm
- Craig Kerwin http://www.icr.com.au/~cdkerwin/microworlds/index.html
- LCSI http://www.lcsi.ca/
- Logo Foundation http://el.www.media.mit.edu/groups/logo-foundation/
- HyperStudio http://www.hyperstudio.com/
- MSW Logo http://www.ultranet.com/~mills

As indicated by Hawkridge (1990), we would all agree that "we want to enrich the existing curriculum and improve the way in which it is delivered, by using computers as sophisticated educational tools which extend traditional ways of presenting information to children".

While The Glennie School still has a great distance to go before truly encouraging a Logo environment in the classroom, we have at least commenced the journey. Both students and teachers have become aware of the turtle and the type of learning it can bring about.

It is important now that we continue in our travels and allow such an environment to be nurtured, not just in the use of MicroWorlds, but in all aspects of our curriculum and thought processes.

References

Hawkridge, D. (1990). Who needs computers in schools and why? In *Computers and Education. 15.* 1-6.

Kerwin, C. (1995). *Supplementary MicroWorlds Teacher's Resources..* Armidale: New Horizons.

Kerwin, C. (1995). *Supplementary MicroWorlds Teacher's Activities..* Volume 1. Armidale: New Horizons.

Kerwin, C. (1995). *Supplementary MicroWorlds Project Starters* Volume One. Armidale: New Horizons.

Papert, S. (1992). *The Children's Machine.* New York: Basic Books.

Spender, D. (1995). *Nattering on the Net.* Melbourne: Spinifex Press.

M.L.C. Merging Microworlds, Minds and Multimedia

Steve Costa
M.L.C.
Kew

Introduction

This paper details the significance of integrating the computer, MicroWorlds and the classroom curriculum.

The importance of identifying areas of the curriculum where the computer can offer students new and better ways of understanding and learning is vital. Teachers and schools are now more aware of the different ways in which students learn. We are also mindful that the whole child must be involved in the learning. No longer can we ignore the theory of 'Multiple Intelligences'. Students should be provided with activities that encourage and promote the use of all their senses to learn, not only a text generated medium. Animation, sound, text and pictures are all vital ingredients in conveying meaning, understanding and the ability to firmly grasp a concept or topic. Students must be actively involved and not just passive observers. They need to construct and create their own knowledge, so it becomes more real and meaningful.

Curriculum: Integrating Content, Delivery and Presentation

Teachers in every year level and in each subject need to be aware of the advantages computers offer in strengthening and reinforcing their areas of study. Computers, however, are often seen as an "extra", as something teachers should use. They are seen as something valuable that students enjoy using; however teachers are often unsure of what they can do. I believe the answer for most teachers is to list the skills they believe are essential. Then list the activities or concepts they want to

develop or build upon. With these points in mind they need to seek "specific" advice from educators who have experience with using computers in educational settings to help integrate their program.

My expectation of the students' capabilities to program using MicroWorlds continues to rise as they constantly impress me with the depth and breadth of their understanding, skill and knowledge. My experience has lead me to believe that students work better and harder when they feel they have a degree of control over their learning. While the computer allows them to construct their own knowledge, they also are eager to learn from their peers. A great deal of sharing and collaboration becomes part of a class that encourages risk taking. I also believe students need to produce a final product. They must see that their efforts are appreciated and are a valuable part of gaining a mark or score towards their report or grade. Too often computer work is seen as only something fun to do and not a vital part of their work. Computers can be used for drill and practice and to assist learning by supplying a great deal of information and knowledge, but they should not only be used as a passive activity.

MicroWorlds and the LogoWriter programs allow students to create and to have control over their learning. But control isn't valuable unless the work being done complements the set aims and objectives of the classroom. It is in the area of curriculum design and implementation that teachers need to spend time searching, creating and developing ways in which the computer can enhance and reinforce learning. I believe gaining new skills in using the computer are important, but even more important is a good imagination that will facilitate sound comprehension and help integrate the skills successfully across a number of learning areas. We do not have a long tradition or years of experience to call upon when using computers in schools. Primary teachers need to be bold and take risks, but must also be aware of the responsibility placed on them to develop the basic skills and knowledge their charges will require. We

cannot ignore our professional responsibilities, and realistically computer skills are now becoming one of the basic skills that will be required of students entering the 21st century.

Creating an exciting and relevant curriculum as well as integrating the computer into the program is not an easy task, or one to take lightly. As the saying goes, it is difficult to serve two masters. However I look upon this challenge as a way to improve the motivation of the students, increase their technological skills and enhance the delivery of the curriculum. A Social Studies unit often forms the basis for the integration of the computer, across a number of subject areas. As an example I have used units on the Solar System, Global Issues and Endangered Species as the basis both for developing computer related skills and highlighting many related subject areas, concepts and bodies of knowledge.

Using a thematic approach, MicroWorlds can be used to create a dynamic report on the Planets and the Solar System. It is also valuable in developing a stronger understanding of number and measurement (distance) related maths activities. Stories and poems about the topic can also be created, with the added advantage of including graphics, animation and sound to the written text. Digital camera images, scanned photos and graphics from other sources help to improve the quality of presentations. Teacher guided and supervised use of CR-ROM reference materials, Netscape and the World Wide Web through the use of the Internet also lead to a better understanding of the available information the students should be interested in and confronted with.

Teachers should try to work in collaborative teams. The need for professional peer supporters as sounding boards, reference resources, mentors and technical advisers is essential for systems to become well established and ongoing programs firmly incorporated into the curriculum. Teachers in the same subject area or year level need to create a learning community within their school or district. The sharing of ideas is important

and each small gain helps to strengthen the entire group. One must remember that it is not essential to create a totally innovative and "high-tech" program to be educationally valid. Small incremental advances are fine. The confidence gained by doing even small computer related activities can be infectious.

Units of study need to be planned. A 6-8 week planner that outlines the skills, new primitives, different procedural methods and the concepts and facts you wish your students to gain and experience is very helpful. In addition, give the students an outline of what is expected. Detail what you want (expect) them to produce and understand. Provide class time to demonstrate the necessary computer skills and commands etc. that they are expected to master and incorporate into their projects. Modelling is important! Teachers will benefit by doing and using the computer procedures, programs and tools they expect the students to use and master. I have gained a greater understanding of the intricacies and potential of the MicroWorlds and LogoWriter software packages by trying to recreate the learning environments and "adventures" I hope the students will experience and discover.

Students often do not know their "limitations" regarding what they can or cannot learn, what they can or cannot create, and understand. We presently have a tool and a medium that can convey and present knowledge in ways we could never have imagined or expected 15-20 years ago. The teacher still needs to provide the stimulus and spark that will spur students on to try to discover the "what if", and create the "what can be", through words, pictures, animation, sounds and ideas that they will understand and accept.

Education is our business and my business in particular, and I believe the use of technology has helped me to do it better. I believe the students will have gained in many ways through using technology and computers in their studies and especially

through programming and creating in a sensory rich medium like MicroWorlds.

- if all student have been able to create something they are proud of
- if the students have felt better about themselves
- if they have gained new skills that they can use in their future studies or work
- if they have been able to understand something better, or
- if they have helped someone else to learn from their efforts,

then part of my ongoing efforts as a teacher has been worth the challenge. I say part, only because one's efforts should never cease nor should one believe one has found the only correct way to impart learning or knowledge. Creating and programming using MicroWorlds has been worth the effort. Its ability to stimulate both teacher and student cannot be undervalued. The capacity to set new horizons and allow for new adventures and discoveries helps to create a community of learners and learning that truly challenges all who enter.

Logo Supporting the P-12 Curriculum in a Technology Immersion School

Anne McDougall
Monash University
Clayton

Jenny Betts
John Paul College
Daisy Hill, Queensland

Acknowledgment

The authors wish to thank the teachers from John Paul College who contributed their time, their ideas and their experiences to the research project on which this paper is based.

Introduction

This paper outlines the ways in which robot turtles, Lego TC logo and MicroWorlds are used to support the curriculum throughout a large P-12 technology-saturated school. It has been developed from data collected to provide a snapshot of the school in 1995, as part of a longitudinal research study of the use of learning technologies in the school. The material here is taken from a set of interviews with teachers and from papers written by teachers: class teachers from the P-7 year levels, and teachers from the faculty areas in the senior part of the school.

Roamer Turtles in P-3 Classrooms

Roamer turtles are used from Pre-school to Year 3 levels, to provide a concrete introduction to Logo ideas. One turtle is provided for every two classes at these levels, so some planning for sharing is needed; however the teachers find that very worthwhile work can be managed with this arrangement.

Planning sessions in which teachers from a year level suggest and discuss ideas together are found to produce good ways in

which the robots can be used to enhance the work students are to do. Most of the turtle activities planned take only a very short time to set up. The children are very keen to use them, and the turtles are rarely out of use.

The turtles' jackets can be covered with butcher's paper, and then decorated by drawing with 'texta' pens or using collage techniques. As well as the decorated jackets, eyes, legs and other attachments can be designed and affixed. "Dressing up" a turtle can involve incidentally many skills from the Art curriculum.

Pre-School

Activities described by teachers from the Pre-school area included reviewing a shopping excursion with the robot visiting cardboard-box shops, and decorating the turtle as a tractor during a Farm Animals theme, using its motion on a model of the farm to study the words "over", "under", "between", and "beside".

Year 1

The Year 1 teacher interviewed described Language benefits from using the turtle, including work with directional and positional language, and with left and right. She also valued turtle work with Mathematical skills of measurement and estimation. Often the turtle is decorated as a focus for parts of thematic work at this level. This teacher described activities integrated into a theme concerning The Sea, including transporting the "Lighthouse Keeper's Lunch", in a basket on the "back" of the turtle, across the classroom to a beach scene, avoiding hanging cardboard replicas of the hungry seagulls described in the story book.

Year 2

The Year 2 teachers emphasised the role of work with the robot turtles as a stepping stone to later work with Logo. They particularly valued student experiences involving critical thinking and problem solving. Sometimes the children follow set programming instructions; at other times they make up their

own sets of instructions for themselves or for others to follow. Imaginative work is undertaken in which the turtle is made into a character, and an environment in which it can move is built. The turtle might be programmed to role play a known story, or the children might make up their own story for the class to listen to while they watch the turtle move.

Year 3

In Year 3 the turtles were used in all curriculum areas, and perceived as an excellent curriculum integration tool. As children develop a program, they record the Logo commands in their books. Sequencing the programming code is seen as an important aspect of the turtle activities. This teacher observed that students use many thinking skills when they program the turtles; in particular they hypothesise a great deal, and test their hypotheses. Mathematical concepts such as measurement, time, spatial awareness, angles and directions are being consolidated by students when the turtles are used, and problem solving strategies are always needed. Also the turtles provide many opportunities for language integration, such as creating students' own stories to match procedures.

One theme treated by the Year 3 children was Creepy Crawlies. The class had been talking about bees and the collection of pollen. To reinforce this Science concept, the children programmed the turtle to visit paper flowers placed around the classroom, to collect pollen. Then it was programmed to bring the pollen back to the hive.

An activity enjoyed very much by these students is story writing with turtle characters. The children "dress up" the turtle to play a character in a story, and create programs to have it walk its way through the plot. This helps students to understand the importance of character development in a narrative. The students work in pairs, with each pair creating a different character; one might be a frog, another a little girl, a third a prince.

Mathematical activities include exploration of acute, obtuse and right angles, and discussion of standard units of measurement

using turtle steps. Estimating seconds, and distinguishing between minutes and seconds are quite difficult for Year 3 children, and the turtle's WAIT command provides the opportunity to work with time.

The Year 3 teacher reported that the children talk a lot about the turtle activities, and there is much sharing of ideas. At this level students have learnt to share the tasks of pressing the buttons, planning the turtle's moves, and so on, and, through the turtle work, have developed good skills of cooperation and listening to one another.

MicroWorlds on Notebook Computers

Students in and above Year 5 have their own notebook computers as well as access to a wide variety of computing and related equipment in the classrooms and the library at the school, so they have essentially unlimited access to computing. MicroWorlds is available on the notebook computers.

Year 5

Language activities with MicroWorlds in Year 5 began with parts-of-speech activities. The teacher wrote selection procedures so students could choose a noun and insert it in a sentence; then they had to select a verb that would work in the sentence, and choose which tense to use. Next the class moved on to some spelling procedures, procedures containing BUTFIRST or BUTLAST commands. The user would have to guess a spelling word with its first or last letter missing, and then type the whole word.

This teacher is observing students developing programming and debugging skills in MicroWorlds. Students at this level often write their own procedures for presentation of ideas or to solve problems. In studying Perimeter, they wrote a procedure that would draw rectangles, and from that developed a formula for perimeter. They studied the code and recognised that the REPEAT 2 in the procedure matched the 2x(length + width) in the formula. These students have used animation to present many

different concepts in Social Studies; for example during their study of the topic Explorers they were given a map of the world and asked to animate the journey of the First Fleet from England to Australia.

Although this teacher does teach some programming directly, a good deal more of the programming is learnt by the students indirectly. The teacher uses new commands and language features in procedures he writes for the students to use. He finds that often commands he has never taught directly appear in the procedures students write.

Year 6

In Year 6 MicroWorlds is used for contract work and projects and for exploring Mathematical ideas such as angles, triangles, squares, and other spatial concepts. Some students write programs for elaborate projects, others are less involved in programming, but move from page to page showing different shapes. Buttons, a title page, and a table of contents are used. Several different activities have been done based on this idea. Another project used animation to show the earth rotating and revolving around the sun, as did a project on the Arctic and another illustrating the water cycle.

Year 7

Year 7 carried out a Mathematics project on shape building. They wrote procedures to make three-dimensional shapes, and to calculate the volumes of these. Problem solving activities were presented to the students on cards from which they could choose; each problem was to be solved by writing a procedure. They also used MicroWorlds to make Mathematics quiz programs, and a Mathematics Dictionary with buttons for the letters of the alphabet, each leading to a page containing definitions of mathematical words to which new words were added continuously throughout the year.

MicroWorlds was used to create an interactive electronic magazine. Graphics were downloaded from Microsoft Works

or Encarta into the MicroWorlds pages. The activity involved a wide range of computing skills, including programming, creating articles for the magazine, composing on-line quizzes, downloading graphics, and so on. Many hypertext ideas, with buttons taking readers into animated parts of the magazine, were used.

After a trip to Canberra, students designed "tourist authority" systems similar to the directory systems sometimes seen in large department stores. The directories contained maps, places to visit, places of interest, historical information, and so on.

Humanities

Humanities classes in the Secondary part of the school use MicroWorlds as an environment in which students explore concepts, definitions and knowledge. For example in a study of Ancient Egypt, students were asked to create a scene illustrating ways in which the Nile was important to the Ancient Egyptians. They drew the river, and created turtles for mud houses, boats, and so on. Then they animated the scene, and had text appear explaining the visuals. Buttons and menus assisted navigation among the pages of information.

Students created pages illustrating their understanding of ozone layer depletion. They drew the Sun, the Earth, and an ozone layer. Rays would travel from the sun, go through holes in the ozone, and then be trapped between the Earth and the ozone layer. Text appeared describing the process and the resulting global warming.

A MicroWorlds timeline was prepared in the context of a unit on Family History. Text, pictures, and sound were used to represent dates in each student's life. Buttons or turtles took the reader to other pages, each of which explained events in a particular year.

Mathematics

Mathematics in the Secondary part of the school uses MicroWorlds extensively. As well as topics such as Shapes and

Polygons, where its use is not surprising, a variety of other areas of the curriculum are supported by programming in this environment. Some examples are described in the following.

For Year 8 students, making a calculator that actually worked was an exciting activity. The students used the paintbrush to create the outline, and turtles to represent the keypad buttons. All students did the basics: addition, subtraction, multiplication and division. Some also created buttons for square and square root. The teacher commented that students took great pride in their creations, and she felt that it was vital that sufficient help with debugging be given for all to achieve success with projects such as this.

The most difficult program attempted to date by the Year 9 students was a Fractions project. The students were given a small start with programming code and were asked to make their work as interactive as possible, including user instructions. The teacher felt that the students gained a great deal of experience from this project as they were able to give it their own individual touches.

Many smaller homework projects were done in MicroWorlds by the Year 9 students. Examples were writing a program to calculate the circumference of a circle or to draw it given the radius, or to calculate the areas and volumes of different shapes.

Worksheets were designed for some projects for the Year 9 students. Teachers assumed that the students should be given the procedures, so they created a file with the required procedures and let each student copy them from the disk. The teachers were fascinated to find that the students preferred not to be given the procedures for these activities; they preferred to take on the challenge of doing their own programming.

The Mathematical work most enjoyed by the teacher and students was a set of projects on fractal geometry undertaken by some Year 10 students. All the fractal activities were initially attempted with pen and paper, before using MicroWorlds, in order to help the students understand iterations and limits. Then

the students typed in the procedures for the fractals and debugged their own programs. They began to experiment and created many different shapes. They worked on the fern, changing the colour of the pen to green, and looked at Pythagorean and Ternary tree procedures, the Sierpinski carpet, and the Mandelbrot set. Some students were very methodical and others just built amazing creations. The teacher is confident they were having more fun with Mathematics than they had ever had before.

Science

A variety of Logo-based activities play important parts in the teaching and learning of Science in the school.

Lego TC logo is used for studies of motion and for design work. Teachers feel that an atmosphere of creativity, exploration and fun exists with this work. It enables them to build on childhood activities, attitudes, and enthusiasm. They believe it improves dexterity, vital for hand-writing, drawing and many spatial activities. It helps students improve understanding of three-dimensional objects, and the connection of these to two-dimensional plans. In design activities in the Science classroom, students compare with pride their creations with those of other students. Confidence gained by demonstrating or explaining a model permeates student attitudes well beyond the Lego TC logo environment. There is also a high emphasis on group co-operation, the ability to listen, to suggest, to instruct, to imitate.

Lego TC logo is considered to be very powerful, and worthy of senior projects in Science; this raises some challenges to satisfy requirements at the senior level for assessment of student work. Approaches such as treating the activities as extended practical exercises in motion studies, or writing diaries including all phases of planning and development as well as scientific precepts used and concepts learned - marked on a criteria fulfilment basis, and with provision to accommodate group and individual contributions - have been used with some success.

Teachers need time to "play" with the MicroWorlds environment, to try out ideas for student activities and to enhance their own programming skills and abilities to express scientific ideas in the medium. MicroWorlds is seen as a superior environment to LogoWriter for scientific work, as its ability to hatch many turtles enables students to program events almost in parallel. The availability of these environments is stimulating a re-think of the assignments that can be set for younger Science students; projects can be designed to stress the processes involved. Some of the programming can be presented as part of the original assignment description. The Science teacher reports that students like this as it gives them a way to start, and the end products of the students' work are still found to be very diverse in nature despite common starting points being provided.

Although many students become competent programmers in this environment, the teacher emphasises that the aim of the work in Science is in fact not to teach programming, but to enable students to utilise the computer as an expressive medium to enhance understanding of scientific ideas and concepts.

Conclusion

From the overview provided here it can be seen that Logo-related activities are being used to support and enhance student work in many curriculum areas, at all levels of the school. Development of sufficient programming capability to use programming as a means to build and express understanding of concepts in various subject areas requires students to have been working in Logo environments for some years. The young children currently playing with the robot turtles are developing powerful ways of enhancing their learning in a variety of curriculum areas, now and in their subsequent school years.

Logo and MicroWorlds at Westall Primary School: A 4 Year Tale

Tonia Chapman
Westall Primary School
Melbourne

Microworlds in the Primary School

The evolution of the microworld, both as a term and a concept, is long and varied (Squires & McDougall, 1986). A microworld is not a single entity nor a prescriptive formula. Primary school children can find effective ways to modularise information and develop their own cognitive strategies in the form of microworlds when presented with a medium such as MicroWorlds or LogoWriter.

A teacher's focus can be based upon a constructionist philosophy (Papert & Harel, 1993), emphasising a non-linear approach in investigation and problem solving. As part of an investigative process children can experience and demonstrate aspects of research, design, production and evaluation as an enriching and rewarding part of their journey through microworlds of learning.

A Brief History 1993 - 1996

The past four years have been evolutionary throughout schools in terms of hardware and software development. Through the teaching of Year 4 children at Westall Primary School , the practice of developing Logo based multimedia projects grew from a fascination with the medium of Logo. This open ended programming language offered potential scope for exhibiting ideas and understandings of children developed through problem solving processes.

Usage of LogoWriter on the Apple IIe was a humble, yet essential starting point that led to an understanding of the complexity and potential of Logo. A box of LogoWriter activity

cards offered a complex resource for a variety of programming approaches and project starters. It is still a useful resource today.

The arrival of new Macintosh computers in the school in mid '93 and the immediate purchase of LogoWriter for the Mac meant more turtles, a wider range of colours and new functions to be explored. In addition to incursions into classical Logo geometry and associated mathematical concepts, personal endeavours studying Logo included investigation into animation and making games that in turn led to development of class projects simulating stories and real life events. The Year 4 tradition of Logo based multimedia projects was "hatched".

During this time period, available literature highlighted the recent development of hypertext style features within Logo environments, addressing the idea of a microworld - a precursor to the MicroWorlds package. In a 1991 publication Harel (1991, p. 390) stated that we can now produce wonderful, interactive, non linear designs.

1993 heralded Logo emerging as such a tool in Grade 4 at Westall.

MicroWorlds became the next essential software item , investigated, researched and documented through personal post graduate study. It entered Westall at an introductory level, a basis for a thesis, later becoming an indispensable tool that again supported and enhanced the work of year 4 children.

Version 2 of MicroWorlds is the newest arrival at Westall, now used in a variety of ways by children in Years 3 - 6, many of whom experienced Logo programming in Year 4. Version 2 has new multimedia features that can take advantage of the new processors in most systems.

ESL Children

Westall Primary School is situated in a highly multicultural suburb in the south eastern part of Melbourne. Ninety percent of the children within the school speak English as a second language (ESL). A majority of the children are from non English

speaking backgrounds (NESB) and speak English as a second or third language, with school being one of the places where children are involved in using English .

Use of MicroWorlds and Logo has been considered as a developmental tool to support these particular children in the following:

- development and practice of communication skills
- learning how to learn
- knowledge development.

Aims and expectations for ESL and NESB children are paced according to individual backgrounds and knowledge. MicroWorlds software , being in part icon driven , has been providing opportunities for non verbal communication that suit a range of children's learning strategies, and styles, (Weir 1987), in early and advanced phases of their English language acquirement and development .

As not all children necessarily learn in a linear pattern according to criteria imposed upon them, (Weir 1987; Goldman Segall 1991; Harel 1991; Satcher 1991), such a pattern of instruction has not been implemented. A constructionist philosophy underpins delivery of experiences for the children. Background knowledge, experience and individuality vary from year to year, and realistic expectations are set in accordance with these.

Classroom Practice

Constructionism as a concept supports the idea of child centred learning. The role of the teacher is seen as active and supportive; learning is considered to be child centred and action oriented - in problem solving contexts.

The success of Logo and MicroWorlds use in Grade 4 lies in moving away from formalised logic-abstract approaches towards intellectual processes that incorporate 'concrete' or constructionist means. Verbal instruction, necessary on an

interpersonal basis, is always complemented with demonstration and hands on involvement.

Children are required to work collaboratively. They are expected to produce information products for the use of other children. This meets requirements of the Technology strand of the Curriculum Standards Frameworks currently in place in Victorian schools. In doing so, these children become totally involved in a process that can be built upon a collection of strategies and ideas that ultimately provide for development of a resulting product, rather like a collaboration of ideas and movement within choreography of a dance.

Students are required to keep a written record of steps and discoveries in a diary format, including comments about their progress with illustrations and code examples, often produced as printed material. Diary or log book keeping is a skill in itself, developing organisational skills and providing a purpose for keeping track of good ideas for further reference. As with many skills, proficiency in this form of writing is a developmental process, and some children are more adept than others, especially when deciding upon appropriate things to write and at an even more basic level finding the language with which to write.

Provision of curriculum support builds a fuller picture of ideas through an integrated curriculum approach, based upon classroom projects incorporating

many subject areas. This information is amalgamated as a basis for the children's projects.

Children's project tasks include:

- manipulation of graphics tools and shape design
- animation of a series of turtle driven shapes through logo programming
- application of logo based primitives in single line commands
- writing procedures to support project functions

- creating text boxes, entering information, using list processing
- sharing and implementing discoveries related to additional multimedia features, including recording of sound and voice, and development of tunes and special effects using the musical keyboard
- writing, reflecting, planning, and sharing
- investigation of LogoWriter style applications, extension of coding applications and testing ideas developed through MicroWorlds and from other sources.

Children's Projects

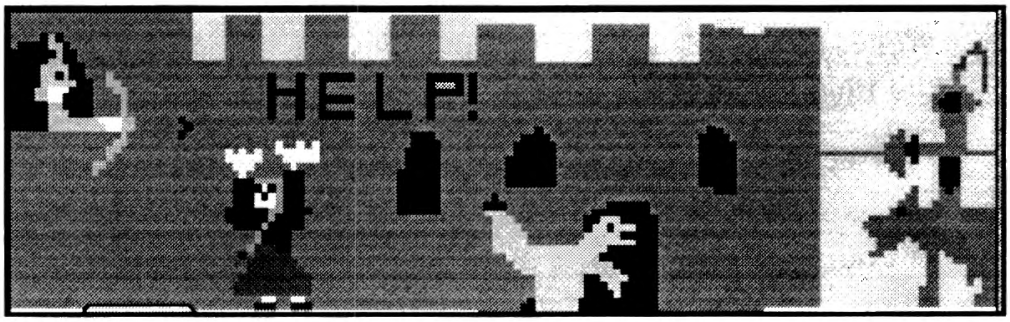
A problem is posed. Children are required to use a number of the MicroWorlds features to implement their ideas. Every year groups of children influence each other in choices and decisions in a number of ways. Certain features of the MicroWorlds program momentarily attain the status of "most popular feature". After intensive exploration of all features, the students settle into their own patterns of working.

A number of children focus upon the graphic nature of their projects. Others insist upon music, sound or voice as a starting point. Another handful want to determine some key functions, establishing movement through direction and heading. Typically, development of class projects loosely follow this routine. Teacher influence also shapes direction of projects.

This year's group prefer structure to get them started, usually based upon information and experiences in other areas of their learning, and through outdoor and indoor games that explore directional and spatial aspects. Procedure writing is workshopped regarding potential problems that could arise later , and during projects when there is need for revision of essential procedures or new discoveries.

In 1996 projects have been developed term by term. Term 1 involved exploration of a few features such as shape animation

and coding buttons to activate shapes. Some ventured into procedure writing. Resulting projects reflected the children's newness in manipulation of Logo language.



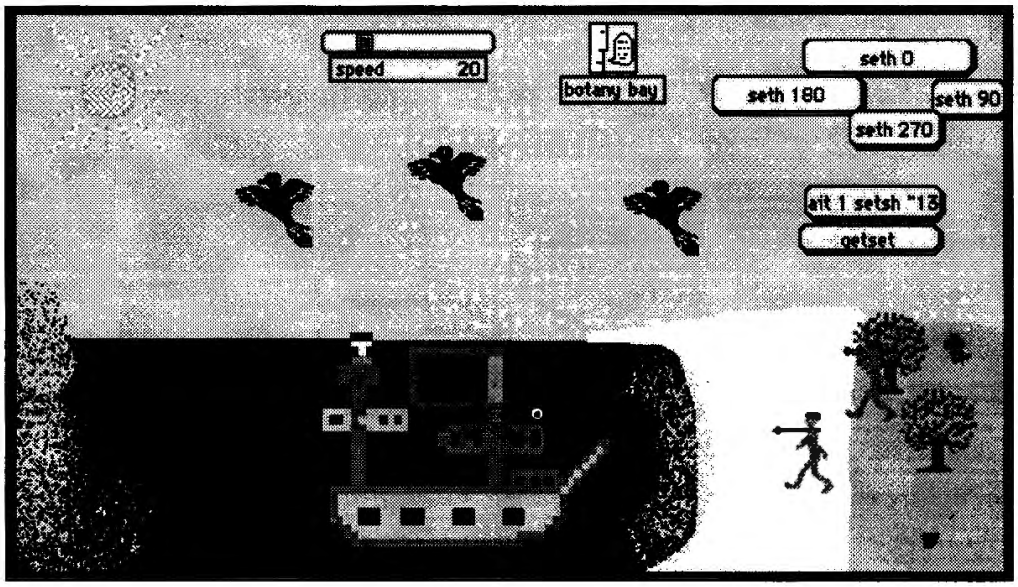
Example of Term 1 Project

Term 2 began with a whole school theme, discovering Australia's past. Projects focussed upon discovery and the arrival of white settlers. Resulting projects generally illustrated a ship arriving at the residence of concerned inhabitants at Botany Bay.

Ships were coded borrowing procedures from MicroWorlds maze activities included within the package. In Term 3 projects the children have demonstrated the ability to reapply a procedure that commands the turtle to change direction after touching a colour. The immediacy of certain procedures supported by MicroWorlds liberate the emergent programmer from some of the hard-to-remember bits of coding found in LogoWriter, such as getting "colorunder" to work properly and remembering how it works when needed in the next project. This is particularly important at Westall P. S. considering language constraints and emerging skills in recording necessary code and other pertinent information.

Term 3 projects have concentrated upon developing increasingly complex projects in which a multiplicity of functions occur. Following the Olympic '96 theme, with a focus upon athletics skills, children chose an athletics event as a basis for exploring animation and procedure writing. Directionality has been further developed, as well as finding ways to explore distance in turtle

steps. Different pairs of children are discovering that coding can occur in more than one way, and are encountering the power of the procedure in contrast to writing a long string of commands.



Example of Term 2 Project

Notable projects of past LogoWriter and MicroWorlds activities include devising ways to present the story of Rapunzel, animating a series of shapes and developing stories or scenes from the life of an invented super hero. Literature themes prove to be a popular starting point.

Future Development

Children at Westall Primary School can continue to build upon their knowledge and experiences developed in Year 4 with increasing complexity in subsequent years, providing that adequate support and information is passed on to other teachers. Resource material is now developing, including children's own resources that they are encouraged to save. The Internet provides further opportunity to share resources and ideas between schools, as do publications such as *LogoFile*.

As the age of electronic information comes to the fore of educational computing, Logo in the form of MicroWorlds stands out as an enduring programming language and communication medium. The near future holds many possibilities - Logo and Microworlds will continue be part of the future in education.

References

Goldman Segall, R. (1991). A Multimedia Research Tool for Ethnographic Investigation. In Harel, I. & Papert, S. (eds.) (1991). *Constructionism - Research Reports and Essays. 1985 - 1990* . Ablex. pp. 467 - 497.

Harel, I. (1991). *Children Designers.* Ablex Publishing Corporation.

Satcher , J. (1991). Different Styles of Exploration and Construction of 3-D Spatial Knowledge in a 3-D Computer Graphics Microworld. In Harel, I. & Papert, S. (eds.) (1991). *Constructionism - Research Reports and Essays . 1985 - 1990 .* Ablex. pp. 335 - 364.

Squires, D. & McDougall, A. (1986). Computer Based Microworlds-A Definition to Aid Design. *Computers and Education, 10,* No . 3 , pp. 375-8.

Weir, S. (1987). *Cultivating Minds - A Logo Casebook.* Harper & Row.

Microworlds: Making the Connections between the Abstract Concepts in Elementary Science

Craig Duncan
Firbank Anglican School
Sandringham, Vic.

Introduction

This paper outlines the benefits of the use of microworlds by elementary students, to explore a number of abstract concepts as found in most science curriculum documents.

Seymour Papert, in 1980, alerted many to the benefits of using microworlds within educational settings. However, he failed to identify sufficiently what is actually required in order to construct such a beneficial learning environment. This paper presents one definition of what a "microworld" is, and what is required to establish and work within them.

At all levels of the Science Curriculum and Standards Framework (Board of Studies 1995), the students are exposed to a number of abstract concepts which can be quite difficult for them to fully comprehend. Two examples of such concepts are the size comparisons between the planets within our solar system and the vast distances that separate them.

The construction of microworlds can not only help the students achieve a greater understanding of such abstract concepts but can also be used to integrate concepts and skills from other curriculum areas. MicroWorlds V2.0, from LCSI, takes both the level of student/microworld interaction as well as the integration possibilities a step further, by providing the students with a variety of tools. These tools can be used to produce multi-media productions through which more comprehensive understandings can be achieved.

In this paper, three specific Level 4 concepts from the Science Curriculum and Standards Framework (C.S.F.) and the way in which microworlds can be created to help make them more "presentable" for students will be discussed. Samples of students work (coding) will be presented to help demonstrate the benefit of using microworlds in this way.

The C.S.F. Based Science Curriculum

As with most science curriculum documents, those which are based upon the Curriculum and Standards Framework (1995) document contain a variety of concepts that are abstract in nature. However, unlike prescribed curricula, those based upon the C.S.F. are essentially constructed to achieve one or more specific student outcomes statements (SOS). This means, quite simply, that provided the outcome is both addressed and achieved, the method(s) that teachers employ to introduce their students to the concept/s, are largely up to their own professional discretion. Papert provides a critique of "traditional" educational settings, by stating:

> *It seems very clear that school gives students a particular model of learning; I believe it does this not only through its way of talking but also through its practices. Skills and the discrete facts are easy to give out in controlled doses. They are also easier to measure. And it is certainly easier to enforce the learning of a skill than it is to check whether someone has "gotten to know" an idea. It is not surprising that schools emphasise learning as "learning that" and "learning how".*
> (Papert 1980 p.136)

The introduction of SOS based curriculum documents provide a positive step towards overcoming the perceptions and weaknesses of "traditional" educational settings, as previously described. Possibly the most important change in pedagogy that has occurred within such curriculum documents is the emphasis placed upon student centred learning, whilst still catering for

different individual learning styles. One way these aspects can be achieved is through the construction of, and exploration within computer microworlds.

Microworlds

In the following discussion the term "microworlds" refers to the learning environment, whereas "MicroWorlds" refers to the computer program from Logo Computer Systems Incorporated (LCSI).

In M*indstorms*, Papert (1980 p.136) explains how "mathetically sophisticated adults use certain metaphors to talk about learning experiences." He further explains how such people describe situations regarding getting to know an idea, exploring an area of knowledge, and how they acquire a sensitivity to distinctions that exist. Papert asserts that children learn in a similar way, except that they are often shown only one model of learning, through their educational settings.

Papert describes how learning in microworlds involves the students establishing a "relationship" with the turtle, through which they are able to "discover facts, make propositional generalisations, and learn skills." However, the emphasis within learning in microworlds is not upon the acquisition of facts nor the rehearsal of skills. Rather the emphasis focuses upon learning the capabilities and limitations of the turtle, through which both a more "informative" and a naturalistic understanding of a variety of concepts can be achieved.

While Papert is able to describe the benefits of learning within microworlds, he fails to adequately describe what is actually required to constitute such a learning environment. David states that a microworld is:

> *a programming environment consisting of, at the very least, (a) a computer and (b) a primitive but extensible vocabulary of words that may be used as building blocks in exploring (c) a particular problem area.* (David 1985 p.13)

This definition clearly relates to Logo, in that Logo is a computer programming environment, it relies primarily upon the use of a "simple" list of primitives - which can be expanded as the users' needs develop, and the majority of the tasks set within the environment have to do with overcoming or solving some problem.

LCSI Mi*croWorlds Project Builder* built upon the capabilities of the Logo programming language by providing a refined and more extensive list of "basic" primitives than those available in previous versions of the language. *Project Builder* also altered the overall appearance of the language by introducing a Graphical User Interface (GUI) to the software in order to increase the level of interactivity that occurs between the user and the language. The GUI of LCSI *MicroWorlds V2.0* has improved again upon the features of *Project Builder* by including a number of other user tools. In addition to the familiar buttons, sliders, and melody editor of *Project Builder, V2.0* includes sound recording capabilities, access to both Quicktime Video and audio CDs for inclusion within projects as well as an ability to interface with LaserDisc technology. In short, as with most other packages available today, LCSI *MicroWorlds V2.0* has a strong multi-media focus. This ensures that truly dynamic and interactive microworlds can be created within the environment.

While the current package is somewhat more "fancy" than previous versions of the language, V2.0 is still primarily Logo. As a result, this has lead to the creation of an environment that is both user friendly and ideally suited for children (and adults!) to explore the concepts to which they are exposed through SOS based curriculum materials.

Addressing Scientific Concepts with Microworlds

The remainder of this paper demonstrates how Year 5 students have both constructed and used microworlds to explore a variety of concepts introduced at Level 4 of the Science C.S.F.

Outcome 1:

> Describe the relationship between speed, distance and time for moving objects.

This outcome statement is introduced at Level 4, under the strand of "The Physical World" in the sub-strand of *Force and Movement.*

Initially, the students in my class were introduced to this outcome statement and the various concepts that it includes by constructing Lego vehicles. These vehicles were then placed upon an inclined plane, released, and their distances and times measured. The students then modified their vehicles to try to improve upon their results.

Following these "hands-on" tasks, the students were asked to write a Logo procedure that demonstrated the changes in velocity that they had observed whilst testing their vehicles. One of my students produced the following microworld:

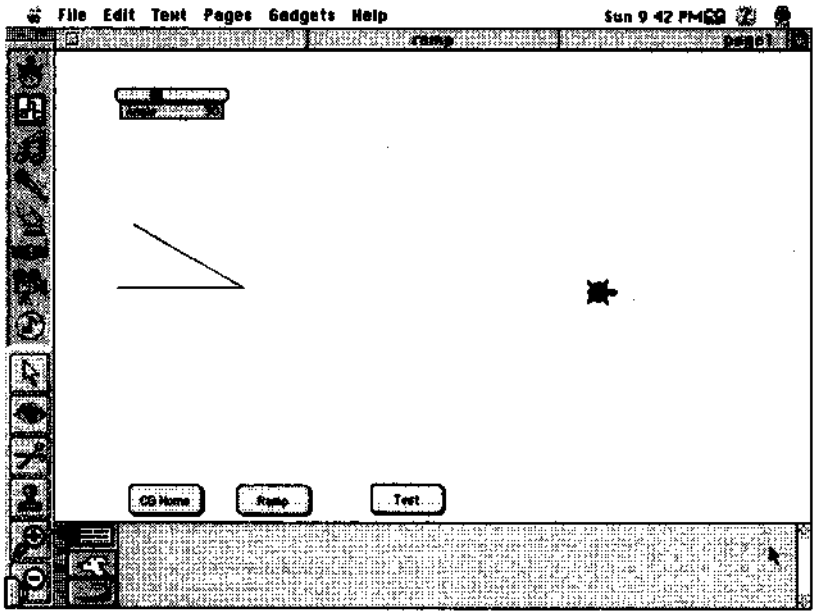


Diagram 1: Screen Display of "Ramp" microworld

```
to ramp
pu setpos [-150 0]
pd lt 90 fd 100 bk 100 rt angle
fd 100
make "s 0
rt 180 pu
end

to test
fd :s
make "y int ycor
if :y < 0 [decrease]
make "s :s + 3
test
end

to decrease
seth 90
make "s :s - 1
fd :s
if :s = 0 [stopall]
decrease
end
```

These procedures demonstrate a clear understanding of both the set task and what was observed during the testing sessions. The procedures draw a ramp, using the value specified by the slider "angle" to indicate the rise. Once the procedure "test" is executed the turtle begins to accelerate down the ramp until it reaches the ground level [ycor < 0] after which de-acceleration occurs. The student also recognised that de-acceleration occurs at a slower rate than acceleration and has included such detail within her programming. While the rates of acceleration or de-acceleration as calculated within the above procedure may make a physicist cringe, I believe that such are more than adequate for a Year 5 student. The above procedure demonstrates that the student has a comprehensive understanding of the concepts

presented and has therefore reached an appropriate standard, as described by the C.S.F.

Outcome 2:

> Design a variety of simple electrical circuits and describe their operation.

This outcome statement is introduced at Level 4, under the strand of "The Physical World" in the sub-strand of *Electricity & Magnetism.*

As with the last statement, the students were introduced to these concepts via hands-on tasks. The students designed and construct various electrical circuits, both in parallel and in series. Voltages were measured in each of the constructed circuits and comparisons were made. The students were once again asked to represent what they had observed by constructing a microworld. Of the various microworlds produced, the following was one of the most comprehensive:

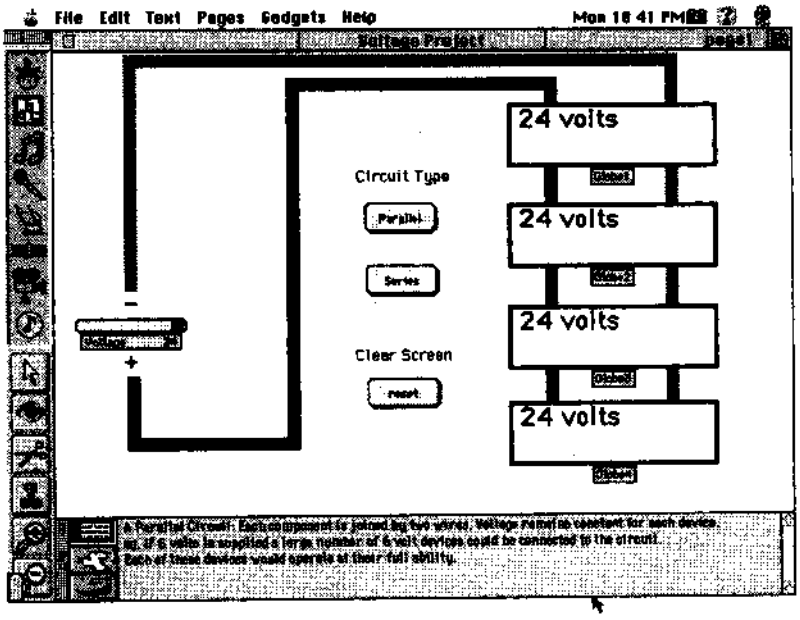


Diagram 2: Screen Display from Circuit Microworld

```
to series
setpensize 10 setc "red ht
pu setpos [-237 0] pd
fd 180 rt 90 fd 390 rt 90 fd 350 rt 90
fd 390 rt 90 fd 90
seriescalc
end

to parallel
setpensize 10 setc "red ht
pu setpos [-237 0] pd
fd 180 rt 90 fd 440 rt 90 fd 300
pu setpos [-235 -79] pd setc "black
fd 50 lt 90 fd 130 lt 90 fd 290 rt 90
fd 210 rt 90 fd 290
paracalc
end

to paracalc
setfontsize 20
make "volt voltage
pd setc "black
tto "Globe1 pr (se :volt [volts])
tto "Globe2 pr (se :volt [volts])
tto "Globe3 pr (se :volt [volts])
tto "Globe4 pr (se :volt [volts])
show (se [A Parallel Circuit: Each component is joined by
two wires. Voltage remains constant for each device.])
show (se [eg. If 6 volts is supplied a large number of 6 volt
devices could be connected to the circuit.])
show (se [Each of these devices would operate at their full
ability.]
end

to seriescalc
setfontsize 20
pd setc "black
tto "Globe1 make "volt voltage * 0.75 pr (se :volt [volts])
```

```
tto "Globe2 make "volt :volt * 0.25 pr (se :volt [volts])
tto "Globe3 make "volt :volt * 0.1 pr (se :volt [volts])
tto "Globe4 make "volt :volt * 0.1 pr (se :volt [volts])
show (se [A Series Circuit: Each component is joined by one
wire. Voltage drops with each device it passes through.])
show (se [For each device to operate at its full potential, the
total voltage of all])
show (se [the devices must be calculated and supplied. ie. 4 x
6v globes would require 24 volts.])
end

to reset
cg ct cc
tto "Globe1 cleartext
tto "Globe2 cleartext
tto "Globe3 cleartext
tto "Globe4 cleartext
end
```

Through her programming the student has displayed a comprehensive understanding of the two types of simple circuits, in terms of both their components and their capabilities and limitations.

The reduction in voltage that occurs between devices within a series circuit was observed by the student, who has attempted to include such detail within her programming (See procedure "seriescalc"). Again, while the reduction rates used may not be entirely accurate, I believe that they are sufficient for an eleven year old child.

Outcome 3:

Describe features of the solar system.

This outcome statement is introduced at Level 4, under the strand of "Earth & Beyond" in the sub-strand of *Our Place In Space.*

The C.S.F. suggests that this outcome has been sufficiently addressed if the student, for example:

- makes a model of the solar system
- compares conditions on the Earth with those on other planets or the Moon and discusses the possibility of setting up a space station on one of them
- researches stars and planets and writes a story about a trip to one of them.
  (C.S.F, p. 46)

One difficulty I have observed with any unit based upon a "space" theme, occurs when discussing with the students the sizes of the planets and/or the vast distances that separate them. This difficulty arises as the students have no frame of reference for such large distances. For example, most children (not to mention adults) have difficulty in comprehending a distance of 142796 kilometres (Jupiter's diameter) or a span equivalent to three and a half times around the Earth.

In an attempt to overcome such difficulties, the students were asked to construct a microworld which would compare the size of any planet against that of the Earth. What follows is one of the microworlds that was constructed.

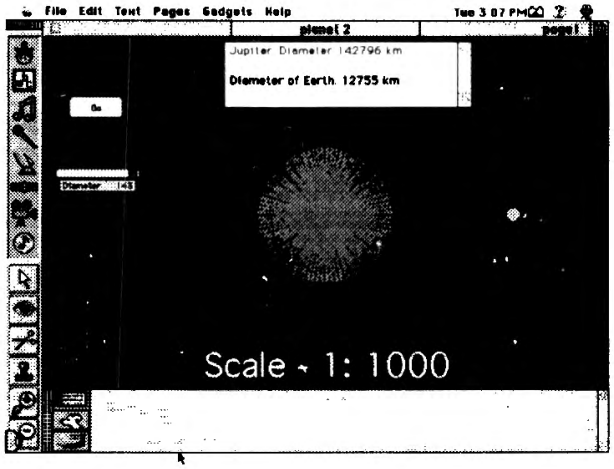


Diagram 3: Planet Comparison Microworld

```
to Go
home ht cg
setbg "black
seth 0
make "radius (diameter / 2)
planetcheck
seth 270 fd :radius bk :radius * 2 home
pu
make "x Heading + 0.5
draw :x
earth
end

to draw :x
pd
seth :x
fd :radius bk :radius * 2 fd :radius
if heading = 180 [stop]
draw :x + 0.5
end

to Earth
setc 65
pu setpos [200 0]
make "radius 6.35
pd
seth 270 fd :radius bk :radius * 2 fd :radius
pu
make "x Heading + 0.5
draw :x
tto "text3 pr [] pd pr (se [Diameter of Earth: 12755 km])
end

to planetcheck
if diameter = 5 [tto "text3 cleartext pr (se [Mercury: Diameter
4878 km]) setc 5 stop]
if diameter = 12 [tto "text3 cleartext pr (se [Venus: Diameter
12104 km]) setc 75 stop]
if diameter = 13 [tto "text3 cleartext pr (se [Earth: Diameter
12755 km]) setc 55 stop]
```

```
if diameter = 7 [tto "text3 cleartext pr (se [Mars: Diameter
6790 km]) setc 15 stop]
if diameter = 143 [tto "text3 cleartext pr (se [Jupiter: Diameter
142796 km]) setc 25 stop]
if diameter = 120 [tto "text3 cleartext pr (se [Saturn: Diameter
120660 km]) setc 45 stop]
if diameter = 51 [tto "text3 cleartext pr (se [Uranus: Diameter
51118 km]) setc 65 stop]
if diameter = 49 [tto "text3 cleartext pr (se [Neptune: Diameter
49528 km]) setc 105 stop]
ifelse diameter = 2 [tto "text3 cleartext pr (se [Pluto: Diameter
2300 km]) setc 85] [cleartext pr (se [Please use the number in
the "Diameter Sets" box only when setting the diameter...])
stopall]
end
```

Through this microworld the student has been able to construct visual, scaled comparisons of each of the planets. Not only does this address the science outcome, but possibly a dozen mathematical ones as well. One difficulty the student observed and commented upon was the limitation of sliders in having the ability to only use integers. As a result the values within the slider needed to be rounded, and therefore the pictorial representations are not truly to scale. (I think I was more impressed with what had been created here than the student was!)

Concluding Comments

Three microworlds have been presented, each of which address a specific outcome statement from Level 4 of the Science C.S.F. Each of the projects could be further developed or refined to allow the investigation of other areas in relation to the same concepts or to permit the exploration of totally new ones.

While only scientific outcomes have been addressed within this paper, microworlds can be created and used to explore nearly any imaginable concept from any curriculum area. Provided that the students are given the three essential elements to construct a microworld, as outlined by David (1985) and that they have a

sound understanding of the basic Logo primitives, then a more comprehensive understanding of any concept, be it abstract or not, can be achieved.

Each of the microworlds presented here has helped the students form more thorough understandings of a variety of abstract concepts, as presented through the C.S.F.. Possibly the most beneficial aspect of creating and using microworlds is that it has allowed the students to explore many of the concepts at a level and rate which is personally relevant and appropriate to them.

References

Board of Studies (1995), Cur*riculum and Standards Framework: Scienc*e, Victoria: Board of Studies.

Papert, S. (1980), M*indstorms: Children, Computers and Powerful Ideas,* Brighton, Sussex: Harvester Press.

David, A. (1985), 'Four instructor roles in an extended Logo environment', A*EDS monitor*, Jan/Feb., pp.13-15.

21 Years of Logo: Logo for the 21st Century

Nicola Yelland
Queensland University of Technology
Kelvin Grove Campus,
Queensland

Jennifer Masters
Queensland University of Technology
Kelvin Grove Campus,
Queensland

Introduction

In the 21 years that we have been using Logo in Australia most of the claims concerning the benefits of Logo made by Papert (Papert 1980) still seem to be relevant. However, even though the type of technology that we use in schools has changed dramatically over this time period, the ways in which teachers use the technology available to them have been disappointing. The research corpus reveals that a large number of studies conducted with and about Logo (Yelland 1995). Initially, there were rich qualitative reports detailing innovative implementations and descriptions of students accessing mathematical ideas in a new way within the technological environment . Then came a series of arguments that focused on attempts to show the effects of Logo via the transfer of skills to other contexts. Fortunately this discussion died a natural death, and more recent research has detailed social and cognitive interactions (e.g. Clements & Nastasi 1985, 1988, 1992; Yelland 1994) within various Logo contexts as well as describing the ways in which students both use and display knowledge of various mathematical concepts and processes (Hoyles & Sutherland 1989; Noss 1988b; Yelland & Masters 1995.

For the past three years we have been working with two versions of Logo called Geo-Logo1 and Turtle Math[2]. Our research has

been based on two aspects of situated understandings. Firstly, the versions of Logo that we have used were specifically designed to support children learning within a curriculum context. Secondly, the content of our work focuses on the importance of the role of the teacher in mediating learning, and the ways in which children may collaborate to solve novel problems within the Logo environment. Geo Logo was developed as part of a large curriculum project in mathematics and forms part of a series of modules which contain activities about specific mathematics topics, such as, paths, co-ordinates and grids and two dimensional shapes. Turtle Math is a stand alone product designed for children in Years 3 to 6 of Primary school.

There are many features of these versions that make them different from those which preceded them. These have been identified elsewhere in detail (Yelland & Masters 1994, 1995), but in summary they are as follows:

1. These versions of Logo establish a dynamic link between the instructions (code) that the child enters and the graphic representations that result from them Any changes in the code, no matter at what stage of the drawing, are immediately reflected by a change in the graphic.
2. The children are supported in their exploration by providing tools that can assist them in problem-solving within the environment. These include not only on-screen protractors, but also features that allow for the modification of the code without having to go back to the beginning of the process or change plans to accommodate a feature which did not turn out as intended.
3. These versions contain features that facilitate the transition from programming in immediate mode to the creation of procedures that can be saved and used at a later time.

Logo: Philosophy and Implementation

With turtle graphics, the idea of programming is introduced

through the metaphor of teaching the Turtle a new word. In the process of programming the locus of control is with the child and not the computer. This is as significant now as it was in the 1980s, because there is still a preponderance of applications where the user is required to adopt a distinctly passive role. From its first implementation, it was apparent that young children liked the turtle, and that they engaged in activities with a high level of motivation and a deep level of concentration. Additionally, their interactions with the turtle enabled them to embark on "powerful kinds of learning" that involved consideration of mathematical content areas such as geometry, applications of number and the operations and algebraic relations, and in some cases they ventured into physics and explored such things as velocities. As Papert (1980) stipulated, "They are learning to speak mathematics, and acquiring a new image of themselves as mathematicians" (p.13). This was at a time when mathematics was often characterised by children as boring, because it tended to involve a) proving that you could "do" addition by repeatedly performing "sums", getting them right and not making careless errors that would scar your record, or b) dealing with abstract concepts that seemed to have no application to life out of school.

We have worked on three very different projects with both versions. In the first instance we trialled the then "new" version with children who were in a Year 3 class. We followed the same children as they worked in pairs into Year 4. This project adopted a case study approach to young children learning with Logo and built up a profile of the children's learning in terms of mathematical understandings, cognitive processes deployed and computer programming abilities. At every stage of the project we scaffolded the children's learning and engaged in conversations in which they articulated their programming strategies and explained particular mathematical processes.

In contrast the second study involved the observation of young children's spontaneous problem-solving in the environment in order to develop a model of performance. We observed 30 pairs

of children over a period of a term. We used the same tasks as in the first study, but only offered technical support to the children as they completed the tasks.

In both studies the children were allocated to pairs by us on the basis of their scores in a non verbal intelligence test; The Coloured Progressive Matrices Test. In the first study there were 7 pairs of children: 2 girl pairs, 1 boy/girl pair and 4 boy pairs. In the second study there were 10 of each gender pair, making a total of 30 pairs of children.

Learning with Logo: Issues and Outcomes

We have noted some interesting observations about the children's involvement with the Logo environment and the activities that were part of it. There are a number of factors that we feel contributed to the quality of the environment in terms of affect and specific performance outcomes.

For example, in the first study we ensured that the environment was collaborative and conducive to problem-solving by:

- encouraging the children to work collaboratively in their pairs, often reinforcing this idea if it was apparent that they were not doing so
- asking them to think, and discuss their plans before they started to direct the turtle, in order to ensure that they might choose the most efficient route
- supporting their exploration with questions about moves that were designed to enable them to reflect not only on their strategies, but also to help them to modify strategies that were not optimal.

In the second study we gave the children instructions for the task that they were about to embark upon, and then left them to their own devices. We told them that we would help if they needed assistance with using the computer.

Some of the differences in outcomes were clear. The children in the first study:

- completed the tasks quickly and efficiently because they were highly motivated
- demonstrated high levels of collaboration in order to solve the task
- spent more time on task, with high levels of concentration
- were able to explain their strategies, to articulate what they were doing and to plan in a very effective manner
- did not become frustrated when things did not go as planned or when they made errors. We were able to support them in their thinking so they could extricate the turtle from the problem situation. This gave them increased levels of confidence and experience so that if the same or similar situation emerged later in the task they were able gradually to figure out the solution independently.

In contrast , we noted in the second study, that the children:

- rarely spent time planning and launched into the task immediately, without consideration or concern about what might be the best way to complete the task requirements
- spent a great deal of time off task
- would leave a task unfinished rather than work through problems or attempt to rectify errors
- did not demonstrate high levels of collaboration. However it was evident that the gender composition of the pair was of importance when considering this aspect of performance, with girl pairs showing more collaborative behaviours than boy or boy /girl pairs
- displayed a sense of frustration and a loss of confidence and motivation for the task when they did not meet the task requirements.

One interesting observation from the second study was in terms of the gender composition of the pairs and task completion in a maze type activity called "get the toys".

The aim of the activity was to take the turtle to the toy and back to the starting position, using the smalest number of commands. For each command that was entered a specific amount of energy was used, and this was indicated by a meter at the top of the game.

When we examined the performance of the gender pairs, the girls were more successful at completing the task requirements than either of the other two gender groupings. Eight girl pairs, five boy/girl pairs and four boy pairs completed the task without running out of energy. A Fisher Exact Probability Test revealed that the only significant difference was between the girl and boy pairs ($p<.10$). Stephens (1992) has suggested that acceptance of this level of probability in a small sample is reasonable since it affords the opportunity to improve the power of the test. This is an interesting finding and would warrant further investigation with a larger sample of children over time and a range of tasks.

When analysing the reasons for the greater success of the girls than the boys it is interesting to note some general trends in performance in terms of the types of interactions and the strategies that the pairs deployed in order to complete the task.

All the boy pairs took route 1, whereas two boy/ girl pairs and one girl pair took a combination of routes 3 and 1, and one boy/ girl pair and two girl pairs took route 3, which was the optimum route in terms of efficiency of moves. The average time for each gender pair to complete the task was very similar: girls - 267 minutes; boys - 301 minutes; boy/ girls - 356 minutes. None of these differences was statistically significant.

Differences among the gender pairs were apparent in the strategies and interactions of the pairs as they attempted to complete the task. In this task it was very important to choose the correct route. If route 1 was chosen, as long as the most efficient moves were made in order to reach the toy, there was just enough energy to complete the task and return to the elevator in 31 moves. A combination of route 1 with 2 or 3 was also possible as long as all moves were combined and reduced to the

minimum. However,many pairs of children who went on this route ran out of energy because they did not combine moves. The most efficient route in terms of energy (route 3) allowed the children some leeway in terms of optimum moves. They could complete the task with 11 moves "spare" which meant that not all moves had to be combined. The two girl pairs who chose this option were very effective in combining moves to make the turtle move the appropriate distance, and completed the task in 23 and 27 moves respectively. Both of these pairs spent time at the start of the task discussing which route they should select and worked it out by counting the dots on the grid as they imagined the turtle travelling along the route. They used their fingers to locate the path that would be followed and then counted one for the distance move and another number for the turn. They also doubled the number when they arrived at the turtle to get the distance for the total trip.

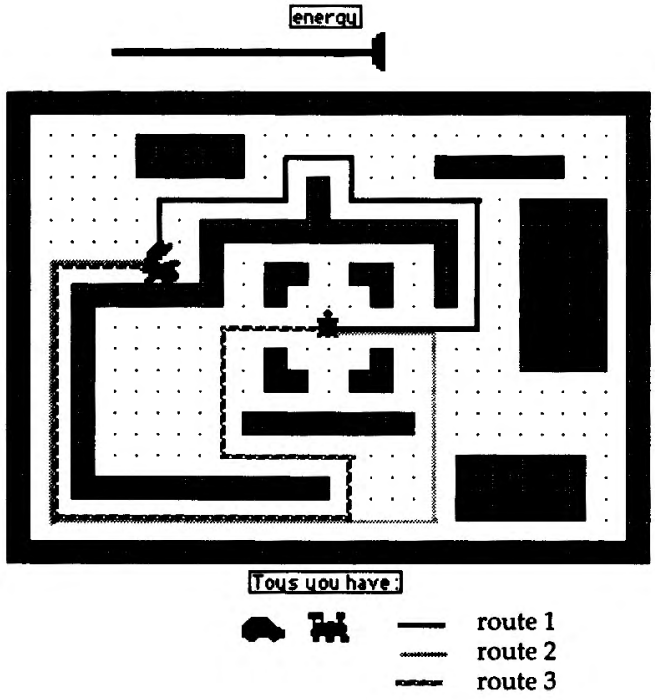


Figure 1. Possible solutions to Get the toys (level 3)

In contrast, all the boy pairs took route 1 and went straight into the task without talking or thinking about which options were available to them. It was important to note that they did this by silent consent, and this was interesting because after the first turn and direction move, it was apparent which route was being taken, and all the boy pairs took alternate turns in order to complete the task.

The boy / girl pairs varied in their approach to the start of the task. In some pairs one member immediately took control and made the first three moves unquestioned, then let the partner have a turn. In others one of the children said' "Let's go this way", and directed the turtle to follow route 1 by making all the moves, only letting the partner gain control of the keys when he or she realised they were in trouble, that is, the energy was getting low as indicated by the message on the computer screen. This style of interaction was not determined by the gender of the child in the pair. In fact, the most interesting observation about the mixed gender pairs was that they did not seem to follow any predictable pattern of behaviour based on gender. In contrast, the girl pairs discussed moves more frequently and seemed not to be concerned about which one actually pressed the keys to effect the move. The boy pairs on the whole regulated the moves by turns, with turns taking place on a regular basis whether this be one, two or five turns before the change.

Conclusions

The new versions of Logo, together with improvements in technology, have made it a language that is both accessible and more powerful for young learners within the context of the mathematics curriculum in schools. With early versions of Logo young children were restricted to programming the turtle in immediate mode and directing it on turtle trips of their choosing as well as making patterns on the screen. Often their exploration of, and with, the language was teacher directed in order to focus on specific mathematical concepts and ideas specific to the curriculum. More recently, versions of Logo have been made

available that empower young children to become sophisticated users of the language. In this way the children that we have worked with have discovered and used complex mathematical notions in authentic and personally constructed contexts, and we hope that this may facilitate later encounters with such abstract mathematical ideas.

Our research has highlighted the role that the teacher may play in such encounters. We have found that the teacher or mediator is essential to focus the child's understandings and to make links, not only with previous experiences in the environment, but also with complex mathematical concepts. In other words the teacher plays a pivotal role in the implementation of technology in the classroom and needs to be aware of the ways in which children learn in order to optimise the learning opportunities for them.

Preliminary analysis of our most recent data indicates that the Logo environments that we have been working with are particularly useful for pairs of girls problem-solving. This is exemplified by the dynamic link between the code and graphic, whereby changes in the former are immediately reflected in the latter. The girl pairs that we have studied are more willing to take risks in such a context in the full knowledge that it will be easy to make changes to rectify problems, a feature not apparent in previous versions of Logo which seemed to inhibit their performance in terms of risk taking and making plans of action.

We feel very positive about the future of Logo as a tool for developing mathematical understandings, in a context that is highly engaging for students and meaningful in terms of engagement with ideas and processes. Our research has shown that young children are able to engage in higher order thinking and use metastrategic (Sternberg 1985) processes. These skills have been identified as being basic skills for the 21 st Century (Blosser 1985). We have come a long way in the past 21 years and now Logo is poised to make a valuable contribution to learning and problem-solving in the 21 st Century as we learn

to integrate technology and varied applications into Primary school curricula.

References

Blosser, P. (1985). Science education for the year 2000 and beyond. In R. K. James & U. R. Kurtz (Eds.), *Science and mathematics education for the year 2000 and beyond* (pp. 52-72). Ohio: School Science and Mathematics Association Inc.

Clements, D. H., & Nastasi, B. K. (1985). Effects of computer environments on social-emotional development: Logo and computer-assisted instruction. *Logo in the Schools, 2*(2/3), 11-31.

Clements, D. H., & Nastasi, B. L. (1988). Social cognitive interactions in educational computer environments. *American Educational Research Journal, 25*(1), 87-106.

Clements, D. H., & Nastasi, B. K. (1992). The role of social interaction in the development of higher-order thinking in Logo environments. In E. DeCorte, M. C. Linn, H. Mandl, & L. Verschaffel (Eds.), *Computer-based learning environments and problem solving* (pp. 229-248). Berlin: Springer-Verlag.

Hoyles, C., & Sutherland, R. (1989). *Logo mathematics in the classroom*. London: R.K.P.

Noss, R. (1988). What is the computer's influence on children's mathematical culture? *Cultural Dynamics, 1*(2), 225-242.

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

Sternberg, R. J. (1985). *Beyond IQ*. Cambridge: Cambridge University Press.

Yelland, N. J. (1994). A case study of six children learning with Logo. *Gender and Education,, 6*(1), 19-33.

Yelland, N. J. (1995). Mindstorms or storm in a teacup? A review of research with Logo. *International Journal of*

Mathematical Education in Science and Technology.., *26*(6), 853 - 869.

Yelland, N. J., & Masters, J.E. (1995). New ways with Logo: Powerful problem-solving for young children. *Quick,*, *54.*,4 - 7.

Yelland, N. J., & Masters, J. E. (1994). Innovation in practice: Learning in a technological environment. In *Australian Association for Research in Education*. Newcastle.

Yelland, N. J., & Masters, J. E. (1995). Learning without limits: Empowerment for young children exploring with technology. In *Australian Computers in Education Conference*. Perth.

[1] Copyright D.H. Clements. Developmental systems, Logo Computer Systems Inc.

[2]1994. D.H. Clements LCSI All rights reserved.

Stars & Sprites

Josie Hopkins
Science Faculty
John Paul College
Daisy Hill, Qld.

Introduction

In the course of this short paper, I wish to relay the experiences of a small number of year 10 students as they grappled with the computer software known as StarLogo, a truly parallel processing version of Logo.

The school culture has a "Notebook computers across the curriculum" programme in place (Grade 5 through Grade 10 in 1996). The computer literacy of the students chosen was somewhat varied, though all have a reasonable grounding in Logo syntax used. Certainly all work on laptops in class daily. Some of the observations made during the course of our using StarLogo are directly reflective of the 'other' learning experiences, expectations and skills of those involved.

An Emergent Choice

The professional educator has been faced with many revolutions in schooling during the twentieth century. From Dewey to Chomsky we have struggled with the best way to present and disseminate knowledge, and struggled with the nature of learning and the vision of childhood itself (Cleaverley & Phillips 1987). But by far the most market driven, the most swift and pervasive change to classroom practice, has been the challenge of embracing computer technology in its fullest capacity as learning tool, and yet ... "The phrase 'technology and education' usually means inventing new gadgets to teach the same old stuff in a thinly disguised version of the same old way" (Papert 1980).

The main impetus for us starting to work in StarLogo (aside from sheer curiosity) was planning for 1997. The students will

be striking larger decentralised systems in their senior science studies, so to be able to demonstrate and test them with a powerful program like StarLogo seems ideal. Due to StarLogo being Macintosh only software, only a small number of students could be involved in the initial investigation. The students were studied as they challenged the assumptions we make regarding complex systems and emergent phenomena, and explored the program. To watch their learning by, through, and in programming the StarLogo environment proved most instructive.

Why Use a Modelling Environment?

Constructionism is the deliberate use of programming to assist students to enhance and develop their own concepts of the world in a manner, at a pace, and in an order which suits each child individually (Harel & Papert 1991). The particular relevance to this project lies in Papert's contention that computers should be used for more than the instructor's fascination with the technology, but rather be an integral part of the learning process itself. Papert argues that the learning process is one involving the construction of the student's own reality. Papert discusses the effective use of the computers in education, in the context of a "window to thinking" (Papert 1985).

"Observation of children's responses in computer-based environments can make their thinking more accessible than it would otherwise be" (Weir 1985). Indeed as a teacher whose central epistemology is constructionism, it has been my experience that students building microworlds, using their favourite software or physical environment, demonstrate their mode of thinking and understanding of concepts with a clarity and finesse otherwise hidden from their assessors.

As with any individualised learning in the classroom, the task set must be carefully chosen so as to be one that might be student centred, indeed student driven, yet embedded in the curriculum. The task focus initiated by the teacher needs to place the central

concept to be studied in a context easily identifiable by the students. This allows the students to develop the ideas with confidence. To provide students with a 'guided start' to a project experience (as occurs with StarLogo in this mini project) is consistent with constructionist epistemology. "You cannot think about thinking, without thinking about thinking about something" (Papert 1985).

Though the consequent directions and tasks undertaken are entirely student driven, the 'creativity strategy' used in this StarLogo project was similar to that suggested by many researchers into learning "... an excellent way to harness the students' understanding for engagement with ideas is to liberate expressiveness" (Lawler 1986).

From an organisational or curriculum perspective, though some direction is afforded by the teacher when using constructionist teaching methodology, the broad programming possibilities when using Logo environments may allow developments only imagined by the students during their work. Thus we appear to have a second level of emergent phenomena to accompany our StarLogo examination of decentralisation ... that of the emergent project!!? The dilemma for the teacher is being able to identify then utilise the speed and directions that students are taking in their studies.

The primary focus of student learning in this project is the insight gained into student understanding of general science concepts, and centralised vs decentralised systems. The study of the "pieces of knowledge" (diSessa 1988) which form the building blocks for the understanding of such complex systems are available in StarLogo. As teacher I am less interested in the syntax of the student programs than I am in their theorising.

> *There are no such things as applied sciences, only applications of science.*
> (Louis Pasteur)

What are Emergent Phenomena?

Our current understanding of complex systems is largely reliant on the concept of a central control mechanism, as we observe certain patterns in nature. The leadership of an ant in an ants' nest, or a lead bird in a flock, or even the 'root' cause of a traffic jam (Resnick, 1992), can be mistakenly interpreted as a centrally controlled system, instead of each entity operating in its own 'universe' and responding to the world as it is designed to do.

Certainly modelling emergent phenomena has long been too difficult for all but the most sophisticated computer programs and complex mathematics algorithms, and therefore the study of any such systems has been considered too hard for younger students. This assumption, however, is in opposition to the observations of children building ideas in the Lego TC Logo environment. Even young students trying to model their own 'amusement park microworld', tend to want "... to run different programs at the same time" (Resnick 1992). To introduce junior secondary students to simple ways of challenging and testing common "centralised" assumptions, opens up new interpretive options for the students in their studies of natural systems.

StarLogo is an (almost) parallel processing language in which one may "hatch" up to 4096 (32^3) turtles at any one time. They can operate as separate entities and will die upon command. Fixed patches add another dimension to the turtles' environment and these too are programmable. The possibilities for the study of natural systems is immense as each individual sprite responds to its environment. Fairly standard Logo programming language is used (with only a few additions and omissions).

Our particular task focus was one of modelling the change of phase of a substance and the Kinetic theories of gases, which is the students' current area of science study. Issues included what is meant by random, what the discussion of average heat of the particles means, probability and discussion of related questions.

The Process

As for any programming language the process for learning StarLogo went something like this ... gidday ~ play ~ ofe ~' dismay ~ OK ~ hooray ~ display.

We recorded the initial 'fooling around' stages which characterise many of the students' navigation around new software for the interest (and considerable amusement) of the students. Students examined work already undertaken by Mitchel Resnick (et al) and the individuals involved with the M.I.T. team. This was followed by the development of some tasks of their own.

Programming investigations began with a simple look at recursive movement and patterns, and then moved on to showing heating substances. The usual programming hitches like defining local and global variables evolved, however the discussions of why and where they needed to be applied enhanced the learning experience rather than compromising it. In particular, the issue of average energy and the degrees of randomness brought forward the issue of the RMS measure for the velocity of particles in a system. According to our Senior Physics curriculum, this is an abstract and most difficult concept, yet it emerged naturally as part of our 'playing' with StarLogo in Grade 10. As with other Logo experiences, the students are embracing 'harder' concepts with ease by controlling the parameters themselves.

The presentation of direct manipulation programs on computer allows the user to model difficult processes and complex questions pictorially. This often reveals the thinking and learning progression of the student (di Sessa 1994). The ability of the teacher to identify misconceptions of their students can allow teachers to modify the learning programme to suit individual needs as required. This presupposes the willingness, and ability of the teacher/facilitator to notice and appreciate the order and nature of tasks undertaken by the students (Lawler 1989). It

also assumes that the teacher has experimented with both the ideas and the program to a degree allowing those sorts of judgements.

You will note that a great deal of time is spent, while working with StarLogo, in the discussion of what is wanted and what is observed from the science perspective. "Sharing a creation can result not only in its refinement, but also in the learner obtaining a deeper understanding of other people's perspectives on the object and on the ideas to which it is related" (Evard 1994). My own bias dictates that I am more concerned with the process than the final project. It is a consistent trend, however, that where the concept has been given time and the impetus to develop, the final project is enhanced. Though a very general statement, this has certainly been our experience with many MicroWorlds and Logo projects in the past.

The major frustration as far as our recent StarLogo experience was concerned, was the lack of opportunity for the students to access the program daily, both in and out of the classroom. For me this distilled the very essence of Logo learning success, the necessity for unlimited access. Other issues raised by the students touched on the lack of text/graphics to which they are accustomed. This, once again, was an the issue of preferred mode of learning/working.

Our current task is one of trying to model electron orbitals (very simply in 2-D) and to display the 'jumping' behaviour and randomness of the same according to random energies of incoming light (photons). A quick overview of atomic theory and then constant discussion and research as we progress, and the results are beginning to 'emerge'. This is an ongoing project. It should be noted here that one of the very useful features of StarLogo is its ability to graph the activities as they occur (e.g. numbers of electrons jumping etc).

Conclusions (Implications for Teaching)

This study using a StarLogo project was carried out at a time

when the students involved were about to embark on their final two years of secondary education. For these students, to better understand their own learning and to be able to build their thinking in many powerful new ways (StarLogo being one), allows them a personal method of viewing difficult concepts, an individual pathway to full understanding. The methodology used in StarLogo allows students to constantly challenge and test what they 'know' and demands that they place their own science learning in context. It allows constant monitoring for modification as required. Our use of this program will be ongoing, and the development of project ideas for next year is an exciting project. But that we might have it on the laptops!

> *It is in creating our own reality that we come to know life in all its complexity and all its glory. It is in identifying the flaws in our privately created universes that we discover our deepest intellectual, emotional and spiritual learning needs ...*

References

Cleaverley, J. & Phillips, D.C. (1987). *Visions of Childhood.* Allen & Unwin: Australia.

Davies, P. (ed.) (1992). *The New Physics.* U.K.: Cambridge Uni Press.

diSessa, A.A. (1985). *Final report on intuition as knowledge.* Report for the Spencer Foundation, Laboratory for Computer Science, Massachusetts Institute of Technology.

diSessa, A.A. (1988). Knowledge in Pieces. In Formen, G. and Pufall, P. (eds.) (1988).*Constructivism in the Computer Age.* Lawrence Erlbaum Associates, pp.49-70.

diSessa, A.A. (1994). What do just plain folk know about physics? To appear in Olson, D.R. (ed.) *Handbook of Education and Human Development: New Models of Learning, Teaching & Schooling.*

Evard, M. (1994). Articulation of Design Issues: Learning Through Exchanging

Questions and Answers. In Y Kafai and M. Resnick (Eds.) (1984). *Constructionism in Practice: Rethinking the Roles of Technology in Learning*. Boston: M.I.T.

Harel, I. (1991). *Children Designers*.. U.S.A.: Ablex Publishing Corp.

Harel, I. & Papert, S. (eds.) (1991). *Constructionism*. 2nd edn. U.S.A.: Ablex Publishing Corp.

Harvey, B. (1996). *Is Programming Obsolete*. WWW, May. bh@cs.berkley.edu

Harvey, B. (1996). *Logo a Capitalist Too*l. WWW, May. bh@cs.berkley.edu

Harvey, B. (1996). *Symbolic Programming vs Software Engineering —Fun vs Professionalism — Are these the same question. WWW*, May. bh@cs.berkley.edu

Heim, M. (1993). *The Metaphysics of Virtual Reality*. U.S.A.: Oxford Uni. Press.

Lawler, R.W. (1984). Designing computer-based microworlds. In Yazdani, M.(ed.) (1984). *New Horizons in Educational Computing*. Ellis Horwood, 40-53.

Lawler, R.W. (1985). *Computer Experience and Cognitive Development*. U.K.: Ellis Horwood.

Levy, S. (1993). *Artificial Life*. U.K.: Penguin Press.

Minsky, M. (1975). A framework for representing knowledge. In Winston, P. (ed.) (1975). *The Psychology of Computer Vision*.McGraw-Hill, pp. 211-277. Reprinted in Haugeland, J. (ed.). (1981). *Mind Design*, MIT Press, pp.95-128.

Minsky, M. (1987). *The Society of Mind*. U.S.A.: Simon & Schuster.

Papert, S. (1980). Mindstorms: Children Computers and Powerful Ideas. U.S.A.: Harper Collins Publishers.

Papert, S. (1985). Computer criticism vs technocentric thinking. *Logo85 Theoretical Papers*. Massachusetts Institute of Technology, pp.53-67.

Papert, S. (1987). *A Critique of Technocentrism in Thinking about the School of the Future*. Paper presented at the Children in an Information Age conference, Sofia, Bulgaria.

Papert, S. (1988). The Conservation of Piaget: The Computer as Grist to the Constructivist Mill. In Forman, G. and Pufall, P. (eds.) *Constructivism in the Computer Age*. Lawrence Erlbaum Associates, pp. 3-13.

Papert, S. (1992). *The Children's Machine*. U.S.A.: Basic Books.

Perkins, D.N. (1991). Technology meets Constructivism: Do They Make a Marriage?, *Educational Technology*, May.

Resnick, M. (1994). Changing the Centralized Mind. *Technology Review*, June.

Resnick, M. (1994). Beyond the Centralized Mindset. *Journal of the Learning Sciences, 5*, no.1 pp.1-22.

Resnick, M. (1994). Learning About Life. *Artificial Life, 1*, no1-2, spring.

Resnick, M. (1995). New Paradigms for Computing, New Paradigms for Thinking. In diSessa, A., Hoyles, C. & Noss, R.(eds). (1995). *Computers and Exploratory Learning*. Springer-Verlag.

Rieber L.P. (1995). A Historical Review of Visualization in Human Cognition. *Education Technology Research and Development, 43* No 1, pp 45-56.

Von Wodke, M. (1993). *Mind over Media*. U.S.A.: McGraw-Hill.

Weir, S. (1984). Logo as an empirical window. In Snorkin, R. (ed.) *Logo84: Pre- proceedings of the 1984 National Logo Conference*. Massachusetts Institute of Technology, 63-75.