After the Gold Rush: Toward Sustainable Scholarship in Computing

Raymond Lister
Faculty of Information Technology
University of Technology, Sydney
Australia
raymond@it.uts.edu.au

Abstract

In just thirty years, we have gone from punched cards to Second Life. But, as the American National Science Foundation (NSF) recently noted, "*undergraduate computing education today often looks much as it did several decades ago*" (NSF, 2006). Consequently, today's "Nintendo Generation" have voted with their feet. We bore them. The contrast between the changes wrought via computer research over the last 30 years, and the failure of computing education to adapt to those changes, is because computing academics lead a double life. In our research lives we see ourselves as part of a community that reaches beyond our own university. We read literature, we attend conferences, we publish, and the cycle repeats, with community members building upon each other's work. But in our teaching lives we rarely discuss teaching beyond our own university, we are not guided by any teaching literature; instead we simply follow our instincts.

Academics in computing, or in any other discipline, can approach their teaching as research into how novices become experts. Several recent multi-institutional research collaborations have studied the development of novice programmers. This paper describes some of the results from those collaborations.

The separation of our teaching and research lives diminishes not just our teaching but also our research. The modern practice of stripping away all 'distractions' to maximize research output is like the practice of stripping away rainforest to grow beef ─ both practices appear to work, for a little while, but not indefinitely. Twenty-first century academia needs to bring teaching and research together, to form a scholarship of computing that is an integrated, sustainable, ecological whole.

Keywords: discipline-based education research, scholarship of teaching and learning, action research.

1 Introduction

The University of Al-Karaouine, Morocco, has existed for over one thousand years. The University of Bologna, Italy, has been granting degrees for over 900 years. Australia has universities that are over 150 years old, and New Zealand's Otago University is almost 140 years old.

Copyright © 2008, Australian Computer Society, Inc. This paper appeared at the Tenth Australasian Computing Education Conference (ACE2008), Wollongong, Australia, January 2008. Conferences in Research and Practice in Information Technology, Vol. 78. Simon and Margaret Hamilton, Eds. Reproduction for academic, not-for-profit purposes permitted provided this text is included.

Universities are among the oldest continuously operating secular institutions in our societies; older than most democratic nations.

A sceptical reader might argue that, while the name is the same, today's universities are profoundly different from the dark-age institutions that perpetuated Aristotle's physics. But even if we accept such an argument, universities remain among our oldest institutions. The modern conception of enquiry-based universities has its origins in Wilhelm von Humboldt (1767–1835) and the associated changes to the nineteenth century German university system. He saw the role of pre-university education as being to present *"closed and settled bodies of knowledge"* whereas universities were to*:*

> *"… conceive of science and scholarship as dealing with ultimately inexhaustible tasks… an unceasing process of enquiry …* [where] *... the teacher does not exist for the sake of the student: both teacher and student have their justification in the common pursuit of knowledge"*
>
> (as described in Clark*,* 1997).

Even if we accept von Humboldt as marking the beginning of universities as we know them, the primacy of research in universities is an even more recent phenomenon. The graduate school only emerged in the last quarter of the nineteenth century (Clark, 1997) as did the use of the very word 'research':

> *The term* [research] *was first used in England in the 1870s by reformers who wished to make Cambridge and Oxford "not only a place of teaching, but a place of learning" and it was later introduced to American higher education in 1906 by Daniel Coit Gilman.* (Boyer , 1990, Page 15)

Prior to World War II, the funding of research in United States universities was not considered a responsibility of the federal government, and it was largely funded by private sources and charities. In 1941, to help with the war effort, universities were mobilized via the creation of the Office of Scientific Research and Development[2]. The National Science Foundation, which today is a source of enormous research funding, did not come into existence until 1951. In 1958, as a direct result of the shock the United States felt from the launch of Sputnik, Congress passed the National Education Defence Act which funded 150,000 new PhDs over the subsequent ten years (Dickson, 2001, p. 227). University research had become big business.

[2] The head of this organisation was Vannevar Bush, best known to computing academics today for 'memex', short for 'memory extender', his pre-computer vision of the World Wide Web (Bush, 1945).

As the post-war research money flowed, a new academic culture grew in which research became the most highly regarded activity within universities ─ possibly the only activity that was highly regarded. This change in American university priorities was reflected in other countries. If the argument is made that universities of one thousand years ago are the same in name only to universities of today, then it also needs to be acknowledged that the post-World-War-II universities are a very new stage ─ and possibly an unsustainable stage ─ in the long evolution of university culture.

1.1 The Boom and Bust of Future Eating

Within the 'green' movement, the term 'future eater' describes a scenario where people over-consume a potentially renewable resource, the resource is lost, and then the people suffer the consequences (Flannery, 1994). I see a pattern of future-eating in the rise and fall of academic disciplines since World War II:

- **Pre-Discipline:** A discipline begins as part of another discipline. For example, computing emerged from mathematics and electronic engineering.
- **Early Boom Discipline:** Enrolment numbers grow to the point where the emerging discipline can sustain an undergraduate curriculum that spans the 3-4 years of an undergraduate degree, either as a major in a more general degree or as a degree in its own right. Universities formally recognize the discipline by creating departments and schools devoted to the discipline. The 1970s were the early-boom years of computing.
- **Late Boom Discipline:** Some students in the undergraduate programme begin to enrol in research degrees, resulting in a large increase in research students. Academics become more focused on their research students than on their undergraduate students, and undergraduate education is increasingly seen as a vehicle to filter for, and prepare, research students. Also, less consciously, academics drift into teaching styles aimed at students who share the academic mindset, which is a small minority of the students. These two forms of disengagement with the bulk of undergraduate students, combined with the burden of large student numbers, leads to poor teaching and widespread student dissatisfaction. This situation can persist for many years, as prospective undergraduates do not see an alternative to studying that discipline. The situation may persist so long that the academics within the discipline come to see it as the natural equilibrium condition, but in fact the discipline is future-eating. In the second half of the twentieth century, physics and English literature were two disciplines that underwent such booms. The 1980s and 1990s were the late-boom years of computing.
- **Bust Discipline:** Prospective undergraduates are attracted to a new early-boom discipline. Enrolment numbers plummet, followed quickly by school and departmental budgets. For example, prospective physics students were drawn to study computing, and prospective English literature students preferred media studies or communications. Just after the turn of the century, computing became a bust discipline.

From the perspective of a university as a whole, the movement of student numbers from one discipline to another discipline is not of vital importance, provided the overall student numbers at the university stay about the same. Today's university operates like a mining company. It exploits the boom discipline until the resource is exhausted, then moves on to the next early boom discipline, leaving the natives of the bust discipline to cope with a degraded environment. To academics inside a discipline, the bust of a discipline can have a profound effect on their academic careers, perhaps even ending their careers. Only academics native to a discipline have a direct interest in the long term sustainability of that discipline.

As the twenty-first century progresses and the world comes to grips with global warming, the short-term industrial mindset of the twentieth century will be replaced with a more environmentally sensitive mindset. In that twenty-first century mindset, the developed world's voracious consumption of resources since World War II, and the social institutions it supported, may be seen as an ephemeral and unhealthy period in human history. If that proves to be the case, then the ephemeral and unhealthy post-World War II priorities of universities will be re-examined, and the relationship between teaching and research will be reconsidered.

Even if it were possible to reverse the trends since World War II and go back to a more peaceful, pastoral, pedagogically oriented academy (and it probably isn't possible), I do not advocate turning back that clock. Instead, in this paper, I will advocate that we adopt a new way of thinking about the relationship between teaching and research. As Rowland (2000) expressed it, my aim in this paper is not so much to "*regain what has been lost – the project of nostalgia – as to write a new story*"(p. 3) I believe that a new story has its basis in two things that have emerged since World War II. The first of these is the concept of ecological sustainability. The second is the development of educational psychology, in forms that can be understood and applied by an academic in any discipline.

But before I can write the new story, I must set the scene by reviewing the old story. In the next three sections I will outline three different conceptions of teaching that are popular today. I will then present an emerging fourth conception of teaching, which has its basis in the ethos of postgraduate research. This new conception does not replace the earlier conceptions. Instead, it completes a set of conceptions that together provide an ecologically sustainable future for computing.

2 Folk Pedagogues I: The Single Institution

When I was a child, my mother was convinced that apple cider vinegar had medicinal powers. Perhaps she was right, as not all 'folk medicines' are without merit. For example, willow bark has been used for centuries as a medicine, and it contains salicin, the active ingredient of aspirin. Folk medicine has been defined thus:

> *Traditional medicine as practiced by non-professional healers or embodied in local custom or lore…* http://medical- dictionary. thefreedictionary.com/folk+medicine

If we replace 'medicine' with 'pedagogy' and 'healers' with 'teachers' then the above quote becomes a workable definition of folk pedagogy:

> *Traditional pedagogy as practiced by non-professional teachers or embodied in local custom or lore…*

The following description of folk medicine has been edited to provide a further description of folk pedagogy:

> *Folk ~~medicine~~* [pedagogy] *... is a category of informal knowledge distinct from "scientific ~~medicine~~* [pedagogy]*" ... is usually unwritten and transmitted orally ...* [and] *... may be diffusely known by many ~~adults~~* [teachers] ... [Folk medicine/pedagogy is] *... not necessarily integrated into a coherent system, and may be contradictory. Folk ~~medicine~~* [pedagogy] *is sometimes associated with quackery ...* [but] *... it may also preserve important knowledge and cultural tradition from the past.*
>
> http://www.windwalkergifts.com/page/page/4185022.htm

Bruner (1996) invoked folk pedagogy to describe our "*intuitive theories about how other minds work*" and that these intuitive theories "*badly want some deconstructing if their implications are to be appreciated*".

2.1 I was a Teenage Folk Pedagogue

Like most computing academics, I began my teaching career as a folk pedagogue. My approach to teaching was a mix of an oral tradition handed down by more experienced colleagues and my own intuitions about what would help the students, which was often a reflection of what had worked for me when I was a student.

Thomas Gray, an eighteenth century British poet, wrote "*Where ignorance is bliss, 'Tis folly to be wise*", and I was indeed blissfully ignorant for several years. My class survey results were good, and I was nominated for a university teaching award. I thought I was a good teacher. As Brookfield (1995, pp. 1-2) expressed it, I was yet another of those academics who "*teach innocently*" and believe that their pedagogical assumptions are "*objectively valid renderings of reality*".

There were two factors that disturbed my bliss. The first was the high failure rates in my class. I taught first year programming, and my failure rates were routinely around 30%, sometimes much higher. In one unforgettable semester, I failed two thirds of the class. I was troubled by these failure rates, but I was able to rationalize the problem by blaming the students ─ their innate ability, their high school preparation and their commitment.

The second disturbance to my bliss was impossible to ignore. First year programming is a politically sensitive area in which to teach, as you must contend not only with your students but also with an intimidating second audience – your colleagues who teach the students in subsequent semesters. Just as I blamed high school teachers for my failure rates, my colleagues blamed me for their teaching problems. In most universities, the academics who teach introductory computing programming are placed under enormous (if well intentioned) pressure by their colleagues. As surely as farmers complain about the weather, computing academics will complain about the programming abilities of students.

I understand why many colleagues who teach upper year electives remain blissful folk pedagogues ─ as long as their students aren't complaining, nobody really knows how well those colleagues are teaching. Had I not taught first year programming, I might also have remained a blissful folk pedagogue to this day.

2.2 The Yoda Retort

The most frustrating forms of intimidation from colleagues are those prefaced with, "*I've been teaching for N years, and …*" where N≥10, usually much larger than 10. I am reminded of a line spoken by Yoda in *The Empire Strikes Back*. Annoyed when Luke Skywalker questions his judgement, Yoda begins his retort "*For eight hundred years have I trained Jedi ...*" To a folk pedagogue, the primary source of knowledge about teaching comes from direct experience of teaching, and therefore the longer someone has been teaching the more they know about teaching. For someone who has been teaching for N<10 years, there is no defence against comments prefaced in this way. Whether it is intentional or not, academics who preface a statement this way are denying legitimacy to the views of their junior colleagues. It is a statement that can only end a discussion.

Justifying a teaching position by citing the number of years that one has been teaching highlights the double life of academics ─ our teaching lives and our research lives. In our research lives, we never justify our position by citing the number of years that we have been researchers. To do so would invite ridicule. Consider your own reactions to the following two assertions:

- "*I have been teaching programming for 30 years, and I tell you students must learn procedural programming before they learn object-oriented programming*".
- "*I have been researching cosmology for 30 years, and I tell you the steady state theory is right*".

2.3 "*When I was a student …*"

For folk pedagogues, their own undergraduate experience is their second source of knowledge, and so another common folk-pedagogic preface is "*When I was a student …*" In my first few years of teaching I drew heavily from memories of my own student days. I copied those teaching techniques my own teachers had used that had worked well on me, and (more commonly) I eschewed those teaching techniques that had not worked for me.

It is legitimate for academics to draw upon their own student experience, but academics should do so with caution. In our undergraduate classes, we future academics were the exceptions to the general rule. When I was a first year undergraduate (almost exactly 30 years ago at the time this paper appeared), I was one of 500 students in my computing class. To the best of my knowledge, I am one of only two students in that undergraduate class who went on to join the academy. It does not follow that what worked well for the two future academics worked well for the other 498 students.

2.4 Denial of a Pedagogic Discourse

The problem with folk pedagogy is not that it inevitably leads to poor teaching. Being a folk pedagogue does not mean that an individual is a bad teacher. On the contrary, and as Kreber (2002, p. 159) expressed it succinctly, "*excellent teachers need not be scholars of teaching*". The problem with folk pedagogy lies at the collective level, not the individual level. A culture of folk pedagogy lacks a mechanism for genuine discourse.

2.4.1 Discourse When We Agree

Without a mechanism for discourse we cannot build upon each other's work. Again, the contradiction is apparent in the double life of academics (i.e. our teaching lives and our research lives). Newton moved physics forward because he "*stood on the shoulders of giants*". In our research lives, we publish our work in the hope that other researchers will build upon it. Thus, the research cycle continues and advances indefinitely, with community members building upon each other's work. In contrast, the folk pedagogue is doomed to reinvent the wheel. Bain (2004, p. 4) illustrated this problem eloquently, when he wrote about the untimely death of a particularly gifted teacher:

> *His colleagues eulogized him, his students remembered his classes, and perhaps a few of them who became teachers carried some pieces of his talent into their own careers. But for the most part his library of teaching talents and practices burned to the ground when he died.*

2.4.2 Discourse When We Disagree

The lack of a mechanism for discourse is an even greater problem when colleagues disagree. Imagine two folk pedagogues, both with equal years of experience, who disagree over some teaching issue ─ how can they resolve their disagreement? They might (and frequently do) regale each other with colourful stories from their respective undergraduate days, but there is no reason why an academic would alter an opinion because of another academic's nostalgic (and possibly romantic) account of their undergraduate experience.

Where there cannot be a resolution of pedagogic disagreement through discourse, office politics will fill the vacuum. In a survey of 31 academics from 19 different computing departments and schools across Australasia, Gruba *et al*. (2004) found that the dominant driving factor in curriculum change was "*influential or outspoken individuals*". In the post-World-War-II academy that promotes staff on the basis of research and not teaching, it does not follow that the most influential or outspoken individuals have greater pedagogical insight.

In response to computing becoming a bust-discipline, many computing degrees have been redesigned. My concern is that most of that redesign has been driven by the intuitions of influential and outspoken folk pedagogues. Consequently, under a veneer of change, largely changes to content, there remain the same old tired pedagogies.

2.4.3 The Silence of Folk Pedagogy

In the previous section I asked (rhetorically) how two folk pedagogues, with equal experience, would resolve a pedagogic disagreement. Most experienced folk pedagogues eventually realize (as I concluded above) that it is futile to attempt to change the minds of fellow folk pedagogues. From that realization emerges a culture of silence. A cynical reader might invoke George Bernard Shaw, who wrote that professions *"are all conspiracies against the laity"* (Shaw, 1906). In fact, a longer extract from that work by Shaw is more illuminating. Shaw was primarily concerned with the medical profession, but (continuing the medical analogy from earlier) his thinking also applies to folk pedagogy:

> *… no doctor dare accuse another of malpractice. He is not sure enough of his own opinion to ruin another man by it. He knows that if such conduct were tolerated in his profession no doctor's livelihood or reputation would be worth a year's purchase. I do not blame him: I would do the same myself. But the effect of this state of things is to make the medical profession a conspiracy to hide its own shortcomings. No doubt the same may be said of all professions. They are all conspiracies against the laity;*

Shaw's observations are less true of medicine today, given the contemporary practice of evidence-based medicine, which applies *"more uniformly the standards of evidence gained from the scientific method"* (Wikipedia, "Evidence-based Medicine") … by using "*the best available external clinical evidence from systematic research"* (American College of Cardiology). However, Shaw's observations remain applicable to folk pedagogy, which does not admit any form of external evidence.

2.5 Evidence

The key to all scholarly discourse is evidence. To be more precise, the key to scholarly discourse is what scholars accept as legitimate forms of evidence. As Rowland (2000) explained:

> *The concept of evidence is arguably the most fundamental concept in all disciplinary enquiry. As the philosopher Jeremy Bentham pointed out, 'the field of evidence is no other than the field of knowledge itself'.* (p. 93)

In the epistemology of the folk pedagogue, there does not exist external evidence; there is just introspection. But if there is to be a non-political resolution to pedagogic disagreement, then some form of external evidence is necessary.

After having taught first year programming for about five years, unable to defend myself by invoking N>10 years of teaching, I commenced a journey, like a latter day Marco Polo, to find what I and others would accept as evidence of good teaching. The journey would take me to three more places, each described in the next three sections, and each with its own conception of legitimate evidence.

3 Folk Pedagogues II: Marco Polo Papers

Valentine (2004) was the first to use an exploration analogy to describe a certain type of pedagogical adventure in computing education. According to Valentine, in the computing education literature a 'Marco Polo' or 'flag planting' paper can be summarized as "*I went there and I saw this*" or "*We tried this and we think it is good*". He explained that the authors of such papers:

> *... describe how their institution has tried a new curriculum, adopted a new language or put up a new course. The reasoning is defined, the component parts are explained, and then (and this is the giveaway for this category) a conclusion is drawn like "Overall, I believe the [topic] has been a big success." or "Students seemed to really enjoy the new [topic]".*

Valentine went on to make the following comment about the usefulness of Marco Polo papers:

> *Marco Polo presentations serve an important function: we are a community of educators and sharing our successes (and failures) enriches the whole community. Yet, it seems that with just a little more effort at ...* [providing evidence] *... we could wring a great deal more benefit from the exercise.*

In the absence of evidence, there is nothing in a Marco Polo paper that can persuade the sceptical reader. Readers of Marco Polo papers can only evaluate the paper against their own teaching experiences and intuitions ─ if the reader is sceptical of the paper's claims, then the reader can safely ignore a Marco Polo paper (e.g. "*It may work with students at that university, but it won't work with my students*."). A Marco Polo paper is folk pedagogy committed to paper.

The potential danger of Marco Polo papers is that they are *'cargo cult research'* (Wikipedia, "Cargo Cult"). That is, the ideas expressed in a Marco Polo paper have the superficial appearance of being research, but the absence of evidence in such a paper means it is not research. However, if a reader finds a Marco Polo paper to be consistent with his/her own folk pedagogy, the danger is that he/she will then cite the paper to colleagues as if it is research evidence. As Mark Twain is purported to have quipped, "*What gets us into trouble is not what we don't know, it's what we know for sure that just ain't so.*"

Valentine found that just over a quarter of CS1/CS2 papers in recent SIGCSE conferences were Marco Polo papers. However, Valentine's classification system probably gives an optimistic impression of the state of the SIGCSE literature, as he does not include as Marco Polo papers other forms of innovation that are also unaccompanied by formal evidence.

Simon (2007) categorized papers from recent ACE and NACCQ conferences, in a way similar to Valentine. However, Simon's categorization included a category of paper, 'report', that included Marco Polo papers and also any paper that describes "*something that has been done, perhaps with a simple survey of student satisfaction*". Simon found that 55% of all recent ACE/NACCQ papers were such reports. From his re-analysis of Valentine's results, Simon concluded that 69% of all CS1/CS2 papers at recent SIGCSE conferences are reports.

In his Ph.D. thesis, Randolph (2007) reviewed a sample of computing education articles published in various conferences and journals between 2000 and 2005. Of the papers that dealt with human participants, Randolph found that 40% provided "anecdotal evidence", a classification that appears to correspond to Valentine's 'Marco Polo' paper (see page 154 of Randolph's thesis).

3.1.1 Hypothesis Generation vs. Confirmation

Randolph (2007) makes the useful distinction between hypothesis generation and hypothesis confirmation:

> *... anecdotal experience has a role in the research process – it has a role in hypothesis generation. But ... there are major problems to using informal anecdotal experience as the sole means of hypothesis confirmation.*
>
> (Page 136)
>
> *... what computer science educators have so far been great at is generating a large number of informed research hypotheses, based on anecdotal experience or on poorly designed investigations. However, they have not systematically tested these hypotheses. This leaves computer science education at a crossroads. To the crossroads computer science education researchers bring a proliferation of well-informed hypotheses. What will happen to these hypotheses remains to be seen. One option is that these informed hypotheses will over time, through repeated exposure* [via Marco Polo papers, reports, and anecdotal evidence] *come to be widely accepted as truths although having never been empirically verified. That is, they will become folk conclusions.*
>
> (Page 176)
>
> *... it makes sense to shift the balance from one that emphasizes anecdotal evidence and hypothesis generation to one that emphasizes rigorous methods and hypothesis confirmation.*
>
> (Page 177)

Once I had realized that Marco Polo papers are about hypothesis generation but not hypothesis confirmation, I recommenced my search for beauty and truth. Eventually I came to another fork in the road ─ was I searching for hypotheses that had already been confirmed by other people's research, or for rigorous research methods with which I would confirm or deny my own hypotheses? A discussion of each of these two forks comprises the next two sections of this paper.

4 Students of Teaching I: The Undergraduate Model

As cited in the introduction of this paper, Humboldt saw the role of pre-university teachers as to present *"closed and settled bodies of knowledge"*, whereas universities were to engage in an education process based on enquiry. Since Humboldt's day, the vast growth in knowledge has resulted in enquiry-based education becoming more the domain of the postgraduate research degree, while the undergraduate degree has become more focused (not

exclusively) on the teaching of relatively closed and settled bodies of knowledge.

After realizing that Marco Polo papers did not provide the evidence for which I was looking, I decided to become a student again ─ a student of teaching. However, was I to be (metaphorically speaking) an undergraduate student or a postgraduate student? That is, should I be seeking authorities who could teach me a *"closed and settled body of knowledge",* or should I engage in an education process based on enquiry?

Every Australian university has, under a variety of names, an organizational unit containing people charged with improving the standard of teaching and learning in that university. I shall refer to such groups as the '*Teaching and Learning group*' (T&L group) and to those groups collectively across Australasia as the T&L community. I shall refer to those who teach computing or any other discipline (e.g. physics, English literature) as '*discipline-based academics*'.

Because discipline-based academics have not provided the second voice required for a conversational discourse on teaching with the T&L community, the 'discourse' has become a one-way transmission of information. T&L groups run staff development programs, where they are the authorities who teach, by a transmission model, their "*closed and settled body of knowledge*" ─ discipline-based academics are taught as if they are undergraduates.

The T&L community have, through the inactivity of discipline-based academics, become the unquestioned authorities on teaching and learning. This is unfortunate, because while they know much that is of value, their recommendations on teaching have weaknesses. In fact, the strength of the T&L community is also their weakness ─ they span disciplines. This is their strength because they are witness to teaching innovation across all disciplines, whereas discipline-based academics are limited to studying innovation in only one discipline. This strength is also their weakness as their ontology emphasizes aspects of teaching that are generic, and thus transportable across disciplinary boundaries. As Rowland (2000, p. 120) expressed it:

> *... a focus on generic approaches to teaching, and theories of learning, can lead to a separation of teaching method and subject matter. Academics or educational developers come to be seen as experts in how to teach but ignorant about what to teach ... like experts of love who have no lover.*

Bowden and Marton (1998, p. 143) expressed a similar sentiment:

> *... being good at teaching means that you are good at teaching something. You cannot teach in general and the way in which you deal with the particular content you are dealing with is what matters.*

Discipline-based academics treasure the knowledge they have of their discipline. Within a discipline, much of the pedagogic discourse is about ways of structuring the knowledge to make it easier for students to understand that knowledge. In contrast, the T&L community tend to play down the importance of discipline-specific knowledge and how to structure it.

The T&L community are a strong proponent of a teaching approach known as constructivism. As a philosophical concept, constructivism has a clear meaning. However, it is not well defined as pedagogy. Different advocates of constructivism-as-pedagogy use 'constructivism' to mean loosely related ideas. Kirschner, Sweller and Clark (2006) describe the approach of constructivist-oriented teachers as follows:

> *First, they challenge students to solve "authentic" problems or acquire complex knowledge in information-rich settings based on the assumption that having learners construct their own solutions leads to the most effective learning experience. Second, they appear to assume that knowledge can best be acquired through experience based on the procedures of the discipline ...*

On the basis of that definition, computing education has used constructivist approaches for decades. For example, many of us introduce students to programming via the problem-solving approach, which McCracken et al. (2001) defined as an approach where we provide students with a problem description, and then require them to decompose it into sub-problems, implement them, test them, then assemble the pieces into a complete solution. As McCracken et al. demonstrated, the problem-solving approach has not proved to be a panacea.

A further difficulty with looking to authorities to provide a closed and settled body of knowledge is that authorities tend not to teach any body of knowledge other than the body they favour. In the next subsection, I will explore Cognitive Load Theory, a theory that I believe offers useful insights into some of the problems we face in teaching computing. However, I have never heard Cognitive Load Theory mentioned at any T&L staff development workshop, perhaps because (unlike constructivism) Cognitive Load Theory places knowledge front-and-centre in its approach to teaching.

4.1 A Brief Tour of Cognitive Load Theory

4.1.1 Knowledge and Long Term Memory

Obviously, an expert in any discipline knows more than a novice in that discipline. However, studies across a number of disciplines (Chi et al., 1988; Ericsson & Smith, 1991) show that experts organize their knowledge in more sophisticated and flexible ways than novices. For example, when asked to memorize board positions in chess, novices were found to memorize the position of each piece in isolation, whereas experts organized the information in terms of the attacking and defensive relationships between the pieces (Chase & Simon, 1973).

In psychological terms, experts are skilful because their long term memories contain huge amounts of relevant knowledge. To most computing academics, the human long term memory might at first appear to be like the long term memory of a computer, but that is a naïve comparison. Unlike computers, a person's long term memory has an uncanny ability to provide, almost instantly, without conscious effort, information relevant

to whatever mental task currently engages the person. Long term memory is so fast and powerful it even changes the way an expert perceives. For example, an expert interpreter of medical X-rays will almost instantly see salient features in an X-ray that a novice might only find after much effort, if at all.

4.1.2 Working Memory and Chunking

Working memory is a concept well-known to many tertiary educated people. Best known is Miller's (1956) result that working memory has a very small capacity ─ seven plus or minus two is the popularly known estimate. Also, working memory can only retain data for about thirty seconds.

Despite these severe limitations on working memory, people's capacity to handle data is greater because of 'chunking'. If a set of associated data items are already stored in long term memory, they may be retrieved and used in working memory as if they were a single item. For example, if someone gives you a new telephone number, retaining that phone number for a few seconds before repeating it will probably consume your entire working memory, as each digit forms one data item to be stored in working memory. However, if you are required to recite, in a specific order, the phone numbers of several friends (and all their phone numbers are in long term memory), then you can do so because each of those familiar phone numbers counts as only one data item in working memory. Thus, the well known limitations of working memory do not apply to all data, but only to data that is not already in long term memory. That is, these limitations to working memory apply to data that has not yet been learned.

4.1.3 Automaticity and Overlearning

A skill can be so well learnt that it becomes 'automatic'. That is, it places little burden on working memory. For example, most car drivers can conduct a conversation with a passenger while driving their car because many aspects of driving have been automated. (Not all aspects of driving can be automated, such as reasoning about traffic at an intersection, where most drivers will cease to speak while they reason about the traffic).

The most common way that skills become automated is through 'overlearning', where a skill is practised long after it has been mastered at a conscious level. For example, professional sports people will practise a specific skill an enormous number of times, until the performance of the skill becomes automatic.

4.1.4 Implications for Teaching and Learning

The concepts of long term memory, working memory, and chunking are important components of Cognitive Load Theory (Sweller, 1999). Within Cognitive Load Theory, learning is described as the process whereby data stored in working memory is transferred to long term memory. Furthermore, from the Cognitive Load Theory perspective, a major instructional problem faced by teachers is the structuring of knowledge so that the working memory of students will not be overwhelmed.

As advocates of Cognitive Load Theory and critics of constructivism, Kirschner, Sweller and Clark (2006) made the following claim:

> *Any instructional theory that ignores the limits of working memory when dealing with novel information or ignores the disappearance of those limits when dealing with familiar information is unlikely to be effective. Recommendations advocating* [constructivist approaches] *during instruction proceed as though working memory does not exist or, if it does exist, that it has no relevant limitations when dealing with novel information …*

4.1.5 The Worked Example Effect

Sweller and Cooper (1985) performed several experiments with students who were studying algebra. In the experiments, some students learnt certain algebraic processes by solving problems, while other students learnt those same algebraic processes by studying worked examples (i.e. students were given a complete solution to each algebraic problem). When both groups were then required to solve more algebraic problems of the same type, the students who had studied the worked examples solved the problems faster, with fewer errors. This finding has been supported in many subsequent experiments in other disciplines, and is known as the *worked example effect*.

Kirschner, Sweller and Clark (2006) explain the worked example effect as follows:

> *Solving a problem requires problem-solving search and search must occur using our limited working memory. Problem-solving search is an inefficient way of altering long-term memory because its function is to find a problem solution, not alter long-term memory. … Thus, problem-solving search overburdens limited working memory and requires working memory resources to be used for activities that are unrelated to learning. As a consequence, learners can engage in problem-solving activities for extended periods and learn almost nothing …*
>
> *In contrast, studying a worked example both reduces working memory load because search is reduced or eliminated and directs attention (i.e., directs working memory resources) to learning the essential relations between problem-solving moves. Students learn to recognize which moves are required for particular problems …*

Cognitive Load Theory may seem simple and obvious in this brief introduction, but applying this theory to teaching is not straightforward. A naïve attempt to construct worked examples might not result in improved learning, as inappropriately constructed worked examples can also impose a high cognitive load on working memory. For example, if a problem is specified as a diagram, and if two spatially separated pieces of information on the diagram need to be integrated by the reader before the problem can be understood, then that integration imposes a load on working memory (Sweller, 1999).

I will return to Cognitive Load Theory in section 5, where it will be used to explain some results in computing education research.

4.2 Teaching and Learning in Perspective

My criticism of the Australasian T&L community position is not that it is an incorrect view of the world. On the contrary, the T&L community know much that is of value. My criticism is that ─ largely due to the absence of discipline-based academics from the discourse on education ─ the T&L community are now positioned in Australasian universities as if they offered a complete explanation of how university teaching should be done. In fact, the T&L community, with their heavy emphasis on non-discipline-specific, constructivist approaches, offer only part of the complete picture.

As mentioned earlier, we (i.e. discipline-based academics) sit in the staff development workshops run by T&L groups as if we were undergraduate students. More than once, I have found it ironic to hear a T&L person complain about their 'undergraduates' (us!), in ways analogous to how discipline-based academics complain about their own undergraduates ─ such as (1) our apparent lack of interest and motivation; (2) an unwillingness to attend lectures (i.e. staff development workshops); and (3) our surface approach to learning (i.e. a tendency to superficially satisfy the university T&L policy without understanding the thinking behind the policy). The Australasian T&L community have these same complaints about their students because they have made the same mistake with their students that we in computing have made with our students ─ they bore us. That is, by presenting pedagogy as a closed and settled body of knowledge, the T&L community guarantee that discipline-based academics will not respect teaching. Enquiry is what academics value and what they are trained to do. To raise the esteem with which teaching is held in Australasian universities, the T&L community need to encourage a culture of enquiry into discipline-based pedagogy. I will discuss an enquiry-based model in the next section.

5 Students of Teaching II: The Postgraduate Model

The difference between being a student of a "*closed and settled body of knowledge*" (i.e. an undergraduate) and being a postgraduate is that the former is about recognizing good answers whereas the latter is about recognizing good questions. Every PhD student learns that lesson, often traumatically. Discipline-based education research is the same[3]. There is no authority to whom we can turn for answers to all the questions we have about our teaching, especially when those questions are specific to our own discipline. From the messy and puzzling world of teaching, we need to formulate a question for which the answer may improve our teaching, and for which we can collect suitable evidence ─ that is, and always has been, the art of research.

[3] In this paper, 'postgraduate' is used as a metaphor. I am not implying that computing education researchers must enrol in PhDs.

5.1 In Defence of Discipline-based Education Research

I find that many of my colleagues in computing will not admit the possibility of computing education research. To them, any form of interaction with students is an aspect of teaching, not a form of research. To those colleagues I reply as follows. As a discipline, we accept that it is legitimate to research the thoughts and actions of people who have graduated from a computing degree (i.e. to study computing as it is practiced in industry), so how can it not be research if we apply the same methods to study the same sort of people before they have graduated? Indeed, how can we hope to develop a comprehensive understanding of the people who have graduated if we do not study that same type of person before they have graduated? The university graduation ceremony is an arbitrary boundary between student and research subject.

Some of my colleagues admit to the possibility of computing education research but believe it should be left to 'the specialists' ─ academics from psychology and education departments. To those colleagues, I reply as follows. Of course those specialists have something to contribute to computing education. However, the study of the path from novice to expert in any discipline, by those already within that discipline, is also a legitimate research programme. Just as a computing academic might not understand some of the subtleties of education and psychology, likewise the education or psychology specialist might not understand some of the subtleties of computing. The computing education researcher should approach educational and psychological theory in two ways: (1) as a platform upon which to elaborate a discipline-specific perspective, and (2) as a general theory that may or may not apply to computing and that needs to be empirically tested by those within computing. Furthermore, to my more philosophically inclined colleagues, I might add that it is a myth that theory ─ especially educational theory ─ comes first, handed down by specialists, followed by application. Education theory often emerges as "*a by-product of the improvement of real situations*" (Carr & Kemmis, 1986, p. 28).

5.2 Adventures of a Middle-aged Postgraduate

In this subsection I will illustrate how we can all profit by studying students from a research perspective. I will do so by showing how my own activity in computing education research helped me to understand a persistent problem in my teaching of novice programmers (and, I believe, a problem that many other teachers also face).

5.2.1 The Problem

For five semesters, from 2002 to 2004, I taught a first-semester programming subject, where the final exam consisted entirely of multiple-choice questions (Lister & Leaney, 2003a&b; Lister, 2005). I adopted that style of exam because it was clear to me that many students could not write code by the end of first semester, and I was tired of setting and marking exams where I pretended that students could write code. My multiple-choice questions fell into two categories:

- '*Fixed code*' questions, where a piece of code was provided and students had to choose from the four

options the value that would be in a particular variable after the code had executed.

- '*Skeleton code*' questions, where students were given a piece of code with one or two lines missing., were told what the code was supposed to do, and were required to choose the correct missing lines of code from the four options provided. These questions were based upon code that I taught in lectures, and consisted of classic sorting and searching algorithms, such as bubble sort, other quadratic sorts, linear search, and binary search. Not only had I taught these algorithms during semester, but I attached to the exam paper a complete set of the PowerPoint slides from lectures, describing these algorithms diagrammatically (over 100 slides). As if providing the lecture notes was not already enough, several weeks prior to the exam, students were provided with a pool of 30-40 multiple-choice skeleton code questions on these algorithms, and they were told that a number of them (usually 5-10) would appear, unaltered, in the exam.

I thought I was setting an easy exam, even though I set the pass mark for the exam at 70%. In fact, in the first semester where I tried this approach, I worried that providing the PowerPoint slides and providing the pool of questions had made the exam too easy. Certainly many of my colleagues thought it was too easy. That first semester, and every subsequent semester, I was astonished to find that the failure rate for the exam was between a quarter and a third of the class, and in the fifth and final semester that I taught the class, the failure rate approached a half.

Why should so many students have trouble answering these questions? For each line of code required by a skeleton-code question, all they had to do was find the appropriate diagram in the PowerPoint slides and effectively turn that diagram into one line of code.

Many of my folk pedagogic colleagues did not share my confusion as to why students struggled with this type of exam. Their explanations were that the students (1) were lazy, or (2) were spending too much time in paid employment, or (3) lacked an essential innate quality, the 'programming gene' or 'geek gene'.

5.2.2 Leeds Working Group

In 2004, I participated in a research experiment with 11 other collaborators from six countries (Lister et al., 2004b). My collaborators arranged to have their own students attempt some of the multiple-choice questions I had used in my exams (but their students did not have prior exposure to similar multiple-choice questions). We collected data from over 600 students. I felt a little better about my own teaching when we found that a quarter of those 600+ students performed at a level consistent with guessing.

Most of the 600+ students answered the multiple-choice questions on paper, or via the web, but we also had a small number of students answer the questions while 'thinking out loud'. Before collecting the 'think out loud' data, I had assumed that students would answer the fixed code questions by the same techniques that I have found that most academics use ─ first, read the code to determine the function performed by the code, then apply the function to the input to calculate the correct answer. I was surprised to find that very few students used that technique. Instead, almost all the students played 'human computer'. That is, they would meticulously hand execute the code to completion. While I had expected that the weaker students might use that technique, I was particularly surprised that the stronger students also solved the questions by playing 'human computer'. Why were novices, especially the better novices, not using the same approach as their teachers? I suspected that the answer to that question was related to the reason why so many of my own students had failed my 'easy' exams.

At the time that we conducted this study, I had taught first year programming almost continuously for 10 years. That I was surprised by the 'think out loud' data demonstrated that I was not learning everything that I needed to know about my students merely through the process of teaching them. To solve my teaching problems, I needed to research my students as well as teach them.

5.2.3 BRACElet

The BRACElet research collaboration began by building upon the findings of the Leeds working group (Whalley, Clear, and Lister, 2007). We prepared a set of exam questions that each project participant then gave to their own students in the end-of-semester exam. One of the questions asked the students to explain 'in plain English' what a short piece of code did. The code was less than 10 lines long (by one common definition of a line of code, there were four lines). The code contained a 'for' loop that iterated over an array, with a single 'if' statement forming the body of the loop. The complete 'explain in plain English' question that we used can be found in another paper in these proceedings (Clear et al., 2008) and elsewhere (Lister et al., 2006a).

When I show this 'explain in plain English' question to academic colleagues, and ask them to answer it, I find that they almost always provide an answer that summarizes the purpose of the code, such as "*it checks to see if the elements of the array are sorted*" (a reasonably correct summary of the code). That answer is not the typical type of answer we get from the majority of students. Instead, students tend to give a (usually correct) line-by-line description of the code. Why were students giving a different type of answer from their teachers? Again, I suspected that the answer to that question was related to the reason why so many of my own students had failed my 'easy' exams.

5.2.4 Related Literature

Since the classic study of chess by Chase and Simon (1973), there have been related studies in many disciplines, including computing (Adelson, 1984; Corritore & Wiedenbeck, 1991; Soloway & Iyengar, 1986; Soloway & Spohrer, 1989; Wiedenbeck, Fix & Scholtz, 1993). The findings from the computing studies have been consistent with the findings for chess and other disciplines ─ expert programmers form more sophisticated and flexible representations of code than novices. The representations formed by experts are based upon the functionality of the code, whereas novices focus more on syntactic features of the code.

5.2.4.1 The Wiedenbeck (1985) Experiment

Wiedenbeck (1985) studied differences between a group of expert programmers and a group of novices. The novices each had about 700 hours of programming experience over two semesters.

The experiment consisted of (1) showing a novice/expert a 'prime', which was a short English descriptive phrase, then (2) showing the subject 1 to 8 lines of code. The code was taken from introductory textbooks on FORTRAN. Each novice/expert had to determine whether the prime was an accurate description of the code. All subjects were shown 108 pairs of prime-then-code. Half the pairs were syntactic (e.g. a prime of 'assignment statement' followed by 'F = F + TOT') and half were functional (e.g. a prime of 'swap two variables' followed by code containing three appropriate assignment statements). Also, half the syntactic pairs and half the functional pairs were consistent (i.e. the prime correctly described the code) and half were inconsistent.

On average, novices made more than twice as many errors as experts, with mean error rates of 8.2% and 3.5% respectively. The speed of response was also measured. On average, novices took almost twice as long as experts to make up their minds, with mean reaction times of 6.1 and 3.2 seconds respectively. The difference between novices and experts was greatest when they were shown an inconsistent functional pair. The mean error rate of novices was 6.3% compared to a mean error rate of only 1.1% for experts. That is, when shown a piece of code that did not correctly implement the specification in the prime, novices did not detect the error 6.3% of the time.

The significance of Wiedenbeck's experiment is, like all research, open to different interpretations. Here is my interpretation. First, we need to bear in mind that the pieces of code the students were asked to read were only 1 to 8 lines long. Second, these students had been learning to program for two semesters, by which stage most of us expect our students to write programs that are a few hundred lines long. Third, an error rate of 6.3% is equivalent to a 50% chance of an error in every ten inconsistent functional pairs ─ if we assume that the code in each inconsistent functional pair is always the maximum 8 lines, then ten pairs comprise (at most) eighty lines of code (and probably a good deal less). Finally, Wiedenbeck's figures are averages, so below average students would have performed worse. My interpretation of Wiedenbeck's experiment is that is not surprising that weaker students, after a year of programming, cannot debug their own programs. Furthermore, the students in Wiedenbeck's experiment learnt FORTRAN, so the number of different programming concepts those students needed to learn and cope with would have been considerably less than the number of programming concepts taught to today's students.

As part of her own interpretation, Wiedenbeck concluded that the expert programmers had to some degree automated the type of tasks she had tested, and this freed the working memory of the experts for higher-level problem-solving tasks. On the other hand, the novices had not automated the tasks, so their working memories were not available for higher-level problem-solving tasks. This led Wiedenbeck to make the following recommendation for teaching programming:

> *For programmers to gain automation it is probably important that the teaching process stress continuous practice with basic materials to the point that they become overlearned. To some extent this goes against contemporary teaching practice.*

By '*continuous practice with basic materials*' I interpret Wiedenbeck as advocating continuous practice with the sort of code segments she used in her experiment, which were only 1 to 8 lines long. I find her experimental evidence persuasive. Given the high failure rates of most first year programming courses, I think we should have tried her suggestion. However, more than 20 years after Wiedenbeck published that paper, her recommendation still '*goes against contemporary teaching practice*'.

5.2.5 True Confessions

From the above research literature, from Cognitive Load Theory, and from the results of the Leeds Working group and BRACElet projects, I have come to see that my 'easy' exams were in based upon folk-pedagogic misconceptions, such as:

- For novices, reading a piece of code is substantially easier than writing a similar piece of code.
- When novices read a piece of code, they quickly abstract to the function of the code
- Novices can move easily between code and diagrammatic (or other) representations of that code.

I have elaborated upon these misconceptions elsewhere (Lister, 2007).

I have taught novices to program for N>10 years … but unlike many who begin a sentence that way, I will finish it by confessing that throughout those N years my folk-pedagogic assumptions were wrong, and today I am embarrassed that I remained blissfully ignorant for so long while students suffered ─ I wish I had developed a research approach to my teaching from the very first day that I began to teach.

It is now 30 years since I was a first year university student learning how to program. Then, and throughout all the years since, failure rates in programming courses around the world have remained high ─ but the folk-pedagogy of programming has remained largely unchanged. We deserved to become a bust discipline.

5.3 Sustainability

This fourth, enquiry-based 'postgraduate' conception of teaching does not supersede the previous three conceptions, but instead completes an ecology that can sustain computing education.

The first three conceptions of teaching perpetuate the pedagogic heritage. Folk-pedagogy, whether it is in the oral tradition or the written 'Marco Polo' tradition, disseminates stories about what appears to work within a discipline. The third conception of teaching, the 'undergraduate' conception, disseminates stories about what seems to be working in other disciplines. These are vital roles.

The fourth conception, the 'postgraduate', with its "*unceasing process of enquiry*" (Clark, 1997), is the conception of teaching that allows us to change. The pedagogy of a fast changing discipline, such as computing, needs to remain intellectually nimble.

I now think that disciplinary knowledge should be organized with learning primarily in mind. This is a fundamental reorientation of disciplinary thinking. For most academics, the undergraduate education is an obstacle course, which only the most worthy students will surmount. I do not mean that we should 'dumb down' computing. I think we must learn to discern between 'dumbing down' computing and making computing more accessible.

To facilitate the process of reorganizing a discipline's knowledge to make it more accessible, I agree with Bowden and Marton (1998, p. 286) that a discipline needs to develop a new research specialization, "*knowledge formation*":

> *The idea is that questions about knowledge formation will be developed into legitimate specializations within* [each discipline]. *This would mean that studies of knowledge formation in physics would become a part of physics proper, for instance, and studies of knowledge formation in social work would become a part of social work proper.*

The "*knowledge formation*" research specialization, through an "*unceasing process of enquiry*", will routinely reorganize disciplinary knowledge, making the discipline more accessible, protecting it from boom-and-bust, and thus sustaining the discipline indefinitely into the future.

6 Thoughts on the Enrolment Decline

As I wrote earlier, many degrees have been redesigned in response to the enrolment decline in computing. My fear is that those redesigns have been driven by folk pedagogues. It is only reasonable that the reader should expect me to say something about the decline, from my perspective as a student of teaching. In this section, I will not attempt to offer up a recipe for reversing the decline ─ it is not the nature of research that it can offer up answers to such big questions when, from a research perspective, the enrolment decline happened very recently. Instead, I shall nominate some literature that I think people should be reading.

Pascarella and Terenzini (1991) wrote the classic book *How College Affects Students*. They surveyed twenty years of research into the American college experience. The following is a frequently quoted excerpt:

> *The research is unequivocal: students who are actively involved in both academic and out-of-class activities gain more from the college experience than those who are not so involved.*

Another classic volume is Seymour and Hewitt's (1997) *Talking about leaving*. It is a study into student attrition from the physical sciences. The reasons students give for leaving might also apply to computing. Among the reasons given are over-packed curricula ('*drinking from a fire hose*') and harsh grading systems.

Tinto (1994) also studied attrition. Among his recommendations for reducing attrition was building a sense of community among students.

All of the above books are written about the American college system. In Australia, Scott (2006) has analysed the written data from the Course Experience Questionnaire (which former students complete within six months of finishing their degree). Scott investigated what factors influenced student engagement. Among his findings was the following (page xvii):

> *"Of particular interest … is the fact that the social affinity subdomain attracted so many hits … This subdomain concerns the nature of the relationships that students experience, not just in the traditional classroom but between peers and with staff from all areas of the university. In short, the CEQuery results strongly suggest that feeling that one's place of study is somewhere where it is great to be has a positive influence on retention …* [and confirms] *… that learning is a profoundly social experience.*

Social affinity is a theme in all of the above literature. I don't believe that computing folk pedagogues are highly conscious of that issue, and consequently I believe it has been largely ignored in recent redesigns of computing courses in Australasian universities. The emphasis in recent redesigns has been on course content, not on the social environment.

Prior to the personal computer, student cohorts developed a sense of camaraderie in the long hours they endured together in the terminal room. I do not propose that we reintroduce those torture chambers, but I think we need to work at building social relationships in today's student cohorts. It is essential that we do that on campus, but I think we could also creatively use web-based social networking technology (i.e. like MySpace), for which this current generation of students already have an affinity.

Some readers may object that the literature I have cited is not concerned directly with the enrolment decline, but is concerned instead with student satisfaction and student attrition. That is a valid criticism. However, I suspect it is very difficult to identify a high school student who will not enrol in computing under current circumstances, but who would have enrolled under different circumstances. Consequently, I believe that all research on the enrolment decline will in fact study proxies for the enrolment decline. For example, when we study attrition, we are studying the students who chose to enrol in computing, not the students who chose not to enrol in computing. In studying attrition, we are assuming that the factors behind attrition are related to the factors behind the enrolment decline. Inevitably, some readers will disagree with my choice of proxies, but equally I think people need to recognize that, for example, research into high school student attitudes to computing is also research into a proxy for the decline.

7 Assorted Asides to Established Researchers

Computing education research remains poorly understood within our discipline. Consequently, most of the space in this paper has been devoted to a manifesto, possibly

polemic, arguing the place and importance of computing education research.

This keynote paper is also an opportunity to share some views with those who are already computing education researchers. This section of the paper is for those people. Each of the following subsections deals with a separate issue. There is no relationship between the subsections.

7.1 The Broader Social Perspective & Funding

The artificial distinction between teaching and research extends beyond computing, and even beyond academia. Our society as a whole makes such a distinction. This leads to government funding models that separate teaching and research, which is illustrated in the following exchange in an Australian Senate Committee (Australian Government, 2007, commencing page 150):

> **Senator CARR** — *If* [the Carrick Institute] *is a research institute, why isn't it in the research division*? [Of the government budget papers]
>
> **Ms Baly** — *It is not, strictly speaking, a research institute. It will undertake research in respect of learning and teaching, but its activities are to promote excellence in learning and teaching within the sector.*
>
> *...*
>
> **Senator CARR** — *... I cannot quite follow, though. It has fellows and various other research grants. Why is it not a research institute?*
>
> **Mr Walters** — *Because it does not just do research. It is about disseminating good practice, you see, as well as carrying out a bit of research. They provide grants in order to do some investigation, and that is intended to support the idea of developing best practice, developing networks about best practice and disseminating the results of that.*

Implicit in the above statements by Ms. Baly and Mr. Walters is a view of university pedagogy as being a closed and settled body of knowledge that needs to be disseminated (i.e. transmitted), but not created.

There is certainly a need for greater dissemination of best practice, and perhaps it is appropriate that the Carrick Institute operate in a way consistent with the above views of Ms. Baly and Mr. Walters. However, we also need a body willing to fund enquiry-based work on education, for academics within a discipline, with an educational focus specifically on that discipline. Such funding bodies exist. For example, the American National Science Foundation has funded computing education research (NSF, 2006). In theory, an academic could go to the Australian Research Council (ARC) for such funding. While an academic in an education faculty may be funded via the ARC, I doubt that an academic in another discipline would be successful in seeking funding for discipline-based education research, ─ the concept of discipline-based education research falls through a gap between the epistemologies of the ARC and the Carrick Institute . Closing that gap may take years, and will happen only if we are patient and continue to work at it.

7.2 On Method

I commend to all computing education researchers the PhD thesis by Randolph (2007), which encompasses a methodological review of computing education research. Randolph's undergraduate education appears to have been in psychology or a related field, so he exhibits a bias for positivist, quantitative research, and sometimes displays a weak appreciation of interpretive, qualitative education research. However, his thesis is an incisive analysis of that part of computing education research that is positivist and quantitative. As part of his thesis, Randolph analysed 93 papers that use an experimental or quasi-experimental approach, and found that most used a one-group posttest-only design. On page 141, Randolph writes that such a design is "*probably the worst of the experimental research designs in terms of internal validity*". On page 164, he elaborates:

> *In the one-group posttest-only design, almost any influence could have caused the result. For example, in a one-group posttest-only design, if the independent variable was an automated tool to teach programming concepts and the dependent variable was the mastery of programming concepts, it is entirely possible that, for example, students already knew the concepts before using the tool, or that something other than the tool (e.g., the instructor) caused the mastery of the concepts.*

Many computing educators are aware of the limitations of the one-group posttest-only design. We tend to adopt that model because our research questions most commonly focus on very large changes to how we teach, and it is therefore not practical to adopt other designs. I do not advocate that we stop researching large changes, but if we also studied smaller changes ─ for example a change to a single class session ─ we might be able to use designs other than the one-group posttest-only design.

On page 177 of his thesis, Randolph advocates that we need to "*shift the balance from one that emphasizes anecdotal evidence and hypothesis generation to one that emphasizes rigorous methods and hypothesis confirmation*". I agree with Randolph. As part of such a shift, I think when we review papers we should all place greater value on papers describing careful, thorough research that is circumspect in its conclusions. Currently, I think we reward authors who make claims that go beyond what can be safely inferred from their results.

7.3 Looking into the Mirror

I recently found myself in conversation with an anthropologist. He explained that today's anthropologists have re-evaluated the work of those pioneering anthropologists from decades past who studied hunter-gathering tribes. Today's anthropologist believes that the thinking of those past anthropologists was so profoundly influenced by their own cultural origins that their research findings were a reflection more of their own culture than of the cultures of the tribes they studied. That conversation led me to wonder whether, in decades hence, computing education researchers would think that today's research on computing students is more a reflection of the researchers than the students.

But why wait for those decades to pass? I think contemporary computing education research should be as much the study of computing teachers as the study of computing students. There are many suitable techniques for studying the teacher (Lewis & Smith, 2005; Lister, et al., 2006b).

7.4 Some Tentative Rules of Discourse

I think we need to work at improving the way that computing education researchers talk to one another. Often, our response to our colleagues is almost folk-pedagogic ─ we compare their finding to our own experiences and intuitions, then accept or reject the work, and the conversation is over. I therefore propose some tentative rules for conducting the discourse of computing education research:

- **The Golden Rule:** Aim to sustain the discourse, not stifle it.

1. Everyone's intuition and personal histories are equally valid when forming hypotheses, and equally invalid when attempting to confirm hypotheses.
2. Sometimes we should debate the evidence, sometimes we should debate what counts as legitimate evidence, and all those present should know which is being debated.
3. Discuss whether the method was used properly, and reported comprehensively, before focusing on the findings ─ especially when the findings appeal to your folk-pedagogy.
4. When there are alternative, equally plausible explanations for the same data, the discussion should focus on further work that will distinguish between the alternative explanations.

8 Conclusion

I have outlined my four conceptions of teaching and learning. Two are folk pedagogical, one with an oral tradition, the other with a written tradition. The two other conceptions are oriented toward teachers as students of learning, one with an 'undergraduate' focus on the study of *"closed and settled bodies of knowledge*" and the other with a 'postgraduate' focus on an "*an unceasing process of enquiry"*.

However, I do not propose that these four conceptions form a hierarchy, with folk pedagogy at the bottom and discipline-based 'postgraduate' research at the top. Computing education research will only ever answer some of the many questions I have about teaching and learning. For those questions that are unanswered by research, I will remain part folk pedagogue, I will continue to read and write Marco Polo papers, and I aim to be the sort of enthusiastic 'undergraduate' that we all love to teach. Not every computing education academic who aspires to be a good computing teacher needs to be a computing education researcher ─ no more than anyone who aspires to be a good programmer needs to design programming languages. What is important is not the conception into which we each choose to (temporarily or permanently) locate ourself. The important thing is two-fold: (1) to be aware of which category we have chosen to locate ourself in, and (2) to remain aware of ─ and to respect ─ the other categories.

The sustained health of 'postgraduate' computing education research depends upon a permanent, two-way relationship with the other conceptions of teaching and learning. From that relationship, 'postgraduate' research will receive a steady stream of research hypotheses, which will then be confirmed or denied via research. I believe a healthy computer science education research programme is based upon a social constructivist view of the world, in which I see "… *the development of theory or understanding as a by-product of the improvement of real situations, rather than application as a by-product of advances in 'pure' theory*." (Carr & Kemmis, 1986, p. 28). All research, including computing education, can become an inward looking, irrelevant, self-indulgent exercise, geared more to furthering careers than to answering important questions. But if computing education research maintains a two-way relationship with the other conceptions of teaching and learning, then computing education research will remain outward looking, focused on answering important questions that improve our teaching.

Metaphors have a profound influence on how we think and subsequently act. Since World War II, the metaphor of the factory and its associated principles of quality assurance have grown to become the principal organizing metaphor of academia. As Rowland (2000, page 7) pointed out, in our teaching lives we 'deliver' our lectures and the 'quality' is 'tracked' and 'benchmarked' by student surveys. Meanwhile, in our research lives, we attempt to increase our research 'capacity' by writing grant applications with clearly defined 'outputs'. In just one twentieth of the time that universities have existed, the post-World-War-II industrial metaphor has degraded the intellectual environment, endangering the health of several disciplines ─ including physics, English literature and recently our own discipline of computing. However, history shows that, so long as the intellectual environment is not entirely extinguished, it responds with vigour when the conditions for growth return. If we change our metaphors, to instead 'cultivate the soil', 'sow the seed', 'tend the field' and 'harvest the crop' … and continue that cycle, all the while remaining sensitive to what the environment can sustain, universities will still be reaping the intellectual harvest in another thousand years.

Acknowledgements

Thanks to Ilona Box and Leslie Schwartzman for their comments on drafts of this paper. Special thanks to Ilona, for her subtle role in helping me develop these ideas over the last 10 years. Without her influence, I might never have become a computing education researcher. I am indebted to Alison Young and Logan Muller who, via inspiring and provocative talks on their work in Peru, started me thinking about academia in terms of ecological sustainability. And I am grateful to Simon for his editing of this paper.

I am an Associate Fellow of the Carrick Institute. However, the views expressed in this paper are solely mine, and not the views of the Carrick Institute.

References

Adelson, B. *When novices surpass experts: The difficulty of a task may increase with expertise*. Journal of Experimental Psychology: Learning, Memory, and Cognition, 10, 3 (1984), 483-495.

American College of Cardiology. *What Is Evidence-Based Medicine*? http://www.acc.org/qualityandscience/quality/evidence.htm [Accessed October 2007]

Australian Government (2007) *Official Committee Hansard SENATE: STANDING COMMITTEE ON EMPLOYMENT, WORKPLACE RELATIONS AND EDUCATION ESTIMATES (Additional Budget Estimates*). February 14, 2007. http://www.aph.gov.au/Hansard/senate/commttee/S9945.pdf

Bain, K. (2004) What the Best College Teachers Do. Cambridge, Mass. : Harvard University Press.

Bowden, J. & Marton, F. (1998) *The University of Learning: Beyond Quality and Competence in Higher Education*. London: Kogan Page.

Boyer, E. *Scholarship Reconsidered: Priorities of the Professoriate*. Princeton, New Jersey: Princeton University Press: The Carnegie Foundation for the Advancement of Teaching, 1990.

Brookfield, S. (1995) *Becoming a Critically Reflective Teache*r, San Francisco: Jossey-Bass.

Bruner, J (1996) *The Culture of Education*, Harvard University Press.

Bush, V. (1945) *As We May Think*. Atlantic Monthly, July.

Carr, W., Kemmis, S. (1986) Becoming critical: education knowledge and action research. Lewes: Falmer Press

Chase, W. C., & Simon, H. A. Perception in chess. *Cognitive Psychology*, 4 (1973), 55-81.

Chi, M. T. H., Glaser, R. & Farr, M. J. (Eds.) (1998) *The nature of expertise*. Hillsdale, NJ, Lawrence Erlbaum Associates.

Clark, B. (1997) The modern integration of research activities with teaching and learning. *The Journal of Higher Education;* 68, 3 (May/Jun).

Clear, T., Edwards, J., Lister, R., Simon, B, Thompson, E. and Whalley, J. (2008) *The Teaching of Novice Computer Programmers: Bringing the Scholarly-Research Approach to Australia*. 10th Australasian Computing Education Conference (ACE2008). Wollongong, Australia. January 22- 25.

Corritore, C. & Wiedenbeck, S. (1991) What Do Novices Learn During Program Comprehension? *Int. J. of Human-Computer Interaction*, 3, 2, 199-222.

Dickson, P. (2001) *Sputnik: The Shock of the Century*. Walker Publishing Co, USA.

Ericsson K, and Smith, J. (Eds) (1991) *Toward a General Theory of Expertise*: *Prospects and Limits*. Cambridge University Press, England.

Flannery, T. (1994) *The future eaters: an ecological history of the Australasian lands and people.* Sydney: Reed Books; New York: G. Braziller.

Gruba, P., Moffat, A., Søndergaard, H., and Zobel, J. (2004) *What drives curriculum change*?. In Proceedings of the Sixth Conference on Australasian Computing Education - Volume 30 (Dunedin, New Zealand). R. Lister and A. Young, Eds. ACM International Conference Proceeding Series, vol. 57. Australian Computer Society, Darlinghurst, Australia, 109-117

Kirschner, P. A., J. Sweller, & R.E. Clark. 2006. Why Minimal Guidance During Instruction Does Not Work: An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. *Educational Psychologist*, 41(2): 75-86. http://www.cogtech.usc.edu/publications/kirschner_Sweller_Clark.pdf [Accessed October 2007]

Kreber, C. (2002). Controversy and consensus on the scholarship of teaching. *Studies in higher education*, 27, 2, pp 151-167

Lewis, T., and Smith, W. (2005) The Computer Science Debate: It's a Matter of Perspective. *SIGCSE Bulletin*. Volume 37, Issue 2 (June 2005) 80-84.

Lister, R and Leaney, J (2003a), *First Year Programming: Let All the Flowers Bloom*, Fifth Australasian Computing Education Conference (ACE2003). Adelaide, Australia. February 4-7. pp. 221-230. http://crpit.com/confpapers/CRPITV20Lister.pdf

Lister, R and Leaney, J (2003b), *Introductory Programming, Criterion Referencing, and Bloom*, 34th Technical Symposium on Computer Science Education (SIGCSE 2003), Reno, Nevada USA, February 19-23, 2003, pp 143-147.

Lister R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, E., Sanders, K., Seppälä, O., Simon, B., Thomas, L., (2004b) A Multi-National Study of Reading and Tracing Skills in Novice Programmers, *SIGCSE Bulletin*, Volume 36, Issue 4 (December), pp. 119-150.

Lister, R. (2005) *One Small Step Toward a Culture of Peer Review and Multi-Institutional Sharing of Educational Resources: A Multiple Choice Exam for First Semester Programming Students*. Seventh Australasian Computing Education Conference (ACE2005). Newcastle, Australia. January 31 - February 3. pp. 155-164. http://crpit.com/confpapers/CRPITV42Lister.pdf

Lister, R., Simon, B., Thompson, E., Whalley, J. L., and Prasad, C. (2006a). *Not seeing the forest for the trees: novice programmers and the SOLO taxonomy*. Proceedings of the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education. (Bologna, Italy, June 26 - 28, 2006). ITiCSE '06. ACM Press, New York, NY, 118-122.

Lister, R., Berglund, A., Clear,T., Bergin, J., Garvin-Doxas, K., Hanks, B., Hitchner, L., Luxton-Reilly, A., Sanders, K., Schulte, C., Whalley, J. (2006b) Research Perspectives on the Objects-Early Debate. *SIGCSE Bulletin*, Volume 38, Issue 4 (December), pp. 173-192.

Lister, R. (2007). *The Neglected Middle Novice Programmer: Reading and Writing without Abstracting*. In the proceedings of the 20th Annual Conference of the National Advisory Committee on Computing Qualifications, NACCQ, Port Nelson, New Zealand,

July 8-11. pp. 133-140. http://site.tekotago.ac.nz/staticdata/papers07/papers/133.pdf

McCracken, M., V. Almstrum, D. Diaz, M. Guzdial, D. Hagen, Y. Kolikant, C. Laxer, L. Thomas, I. Utting, T. Wilusz, (2001): A Multi-National, Multi-Institutional Study of Assessment of Programming Skills of First-year CS Students. *SIGCSE Bulletin*, **33**(4):125-140.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63, 81–97.

NSF (2006) CISE Pathways to Revitalized Undergraduate Computing Education (CPATH) http://www.nsf.gov/funding/pgm_summ.jsp?pims_id=500025 [October 2007]

Pascarella, E.T. and Terenzini, P.T. (1991). *How college affects students: Findings and Insights from Twenty Years of Research.* San Francisco: Jossey-Bass.

Randolph, J. (2007) "Computer science education research at the crossroads: A methodological review of the computer science education research: 2000-2005". PhD dissertation: Utah State University. http://www.archive.org/details/randolph_dissertation [Accessed October 2007].

Rowland, S. (2000) *The enquiring university teacher*. Buckingham: SRHE and Open University Press.

Scott, G. (2006). *Accessing the Student Voice - Using CEQuery to identify what retains students and promotes engagement in productive learning in Australian higher education*. Barton, Australian Capital Territory: Department of Education, Science and Training. http://www.dest.gov.au/sectors/higher_education/publications_resources/profiles/access_student_voice.htm

Seymour, E., and Hewitt, N. (1997) *Talking about leaving: why undergraduates leave the sciences*. Boulder, CO: Westview Press.

Shaw, G. B. (1906*) The Doctor's Dilemma: Preface on Doctors*. http://www.gutenberg.org/dirs/etext04/dcprf10.txt [Accessed October 2007]

Simon (2007) A Classification of Recent Australasian Computing Education Publications. *Computer Science Education*. Vol. 17 Issue 3, (Sep) p155-169.

Soloway, E. and Iyengar, S., Eds *Empirical Studies of Programmers*. Ablex, NJ, USA,1986.

Soloway, E. and Spohrer, J. (Eds) *Studying the Novice Programmer*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1989.

Sweller, J. (1999) *Instructional Design*. Camberwell, Victoria: ACER Press.

Sweller, J., & Cooper, G. A. (1985). The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction*, 2, 59–89.

Tinto, V. (1994) *Leaving College: Rethinking the Causes and Cures of Student Attrition*. University Of Chicago Press; Second edition.

Valentine, D. (2004). CS educational research: a meta-analysis of SIGCSE technical symposium proceedings. In Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education (Norfolk, Virginia, USA, March 3-7). SIGCSE '04. ACM Press, New York, NY, 255-259.

Whalley, J, Clear, T, and Lister, R. (2007) The Many Ways of the BRACElet Project. *Bulletin of Applied Computing and Information Technology* (BACIT) Vol. 5, Issue 1. ISSN 1176-4120. http://www.naccq.co.nz/bacit/ 0501/2007Whalley_BRACELET_Ways.htm

Wiedenbeck, S. 1985. Novice/expert differences in programming skills. *International. Journal of Man-Machine Studies,* 23, 4 (Oct. 1985), pp. 383-390.

Wiedenbeck, S., Fix, V. & Scholtz, J. (1993) Characteristics of the mental representations of novice and expert programmers: An empirical study. *International Journal of Man-Machine Studies*, 39, 793-812.

Wikipedia. *Cargo Cult* http://en.wikipedia.org/wiki/Cargo_cult [Accessed October 2007]

Wikipedia. *Evidence-based Medicine* http://en.wikipedia.org/ wiki/Evidence-based_medicine [Accessed October 2007]