Conferences in Research and Practice in
Information Technology

Volume 78

Computing Education 2008

Australian Computer Science Communications, Volume 30, Number 5.

AUSTRALIAN
COMPUTER
SOCIETY

Computing Education 2008

Proceedings of the Tenth
Australasian Computing Education Conference (ACE2008),
Wollongong, NSW, Australia, January 2008

Simon and Margaret Hamilton, Eds.

Volume 78 in the Conferences in Research and Practice in Information Technology Series.
Published by the Australian Computer Society Inc.

Published in association with the ACM Digital Library.

Computing Education 2008. Proceedings of the Tenth Australasian Computing Education Conference (ACE2008), Wollongong, NSW, Australia, January 2008

Conferences in Research and Practice in Information Technology, Volume 78.

Editors:
Simon
School of Design, Communication and Information Technology
University of Newcastle
Ourimbah Campus
PO Box 127
Ourimbah
NSW 2258
Email: Simon@newcastle.edu.au

Margaret Hamilton
Computer Science and Information Technology
RMIT University
Melbourne, Australia
Email: margaret.hamilton@rmit.edu.au

Series Editors:
Vladimir Estivill-Castro, Griffith University, Queensland
John F. Roddick, Flinders University, South Australia
Simeon Simoff, University of Technology, Sydney, NSW
crpit@infoeng.flinders.edu.au

Publisher: Australian Computer Society Inc.
PO Box Q534, QVB Post Office
Sydney 1230
New South Wales
Australia.

Conferences in Research and Practice in Information Technology, Volume 78.
ISSN 1445-1336.
ISBN 978-1-920682-59-0.

Printed, November 2007 by Flinders Press, PO Box 2100, Bedford Park, SA 5042, South Australia.
Cover Design by Modern Planet Design, (08) 8340 1361.

The *Conferences in Research and Practice in Information Technology* series aims to disseminate the results of peer-reviewed research in all areas of Information Technology. Further details can be found at http://crpit.com/.

Table of Contents

Preface

Welcome to the Tenth Australasian Computing Education Conference (ACE2008), held as part of Australasian Computer Science Week 2008 in Wollongong, Australia. This year's ACE promises to continue the tradition of a high quality conference providing the opportunity for educators from all areas of computing to meet and share their research and innovations in computing education.

The full papers presented in these proceedings represent a good cross-section of leading-edge developments in the field of computing education research and innovation. An international call for papers resulted in the submission of 39 papers from Australia, Costa Rica, Finland, Germany, Jordan, New Zealand, Sweden, the UK and the USA. Each paper was refereed double blind by three or four referees, and 18 papers (46%) were finally selected for publication and presentation.

The chairs are extremely grateful to all the members of the program committee and to the additional referees who gave their time and expertise to this process, sometimes at the cost of being ungratefully hounded for their reports. Part of the reason for this is that there were rather fewer referees than for recent past conferences, and they were therefore asked to referee more papers than they had become used to. We urge participants in ACE2008 to consider offering their services as referees for ACE2009, and to encourage suitable colleagues to do the same.

This year's keynote paper is presented by Raymond Lister, one of the leading lights of the Australasian computing education community. For this paper Raymond has crystallised his thoughts on the past, present, and future not just of computing education research but of computing education as a university discipline, and we expect his plenary presentation to provoke serious thought among all of the academics attending ACSW.

Thanks to ACM SIGCSE, we also have an invited paper – a re-presentation of a paper that has already been presented at a SIGCSE conference. With financial assistance from SIGCSE, Anders Berglund has come from Sweden to present a paper that first appeared at ITiCSE in 2006; and with permission from ACM, we reproduce that paper in these proceedings.

Pre-conference workshops have been a feature of ACE since 2004, and the tradition continues and expands this year with a two-day pre-conference workshop on the classification of computing education papers and a one-day end-of-conference workshop on the continuing BRACElet study of novice programmers. These workshops are a wonderful opportunity for members of the computing education community to work together on a joint project that almost invariably leads to one or more publications.

This year saw ACE catch up with much of the rest of the conference world in the adoption of a web-based paper submission system: we have used EasyChair, and are happy to recommend it to other conference chairs. This year, too, the chairs undertook to (sub-)edit about half of the papers, whose written expression was less than ideal even after rewriting and resubmission by the authors. We believe that this has significantly raised the standard of the proceedings, and are considering how we might make such editing a standard feature of future ACEs.

We thank CORE and everyone involved in Australasian Computer Science Week for making this conference and publication possible, and we thank the Australasian Computing Education Executive for the opportunity to chair this conference.

Simon
University of Newcastle, Australia

Margaret Hamilton
RMIT University
ACE 2008 Programme Chairs
January 2008

Programme Committee and Review Panel

Programme Committee

Simon, University of Newcastle, Australia (senior co-chair)
Margaret Hamilton, RMIT University, Australia (co-chair)

Angela Carbone, Monash University, Australia
Tony Clear, Auckland University of Technology, New Zealand
Mats Daniels, Uppsala University, Sweden
Michael de Raadt, University of Southern Queensland, Australia
Sally Fincher, University of Kent, UK
John Hamer, Auckland University, New Zealand
Judy Kay, University of Sydney, Australia
Raymond Lister, University of Technology Sydney, Australia
Anthony Robins, University of Otago, New Zealand
Judy Sheard, Monash University, Australia
Josh Tenenberg, University of Washington, USA
Alison Young, Unitec, New Zealand

Additional Reviewers

Dave Bremer, Otago Polytechnic, New Zealand
Martin Dick, RMIT University, Australia
Alan Fekete, University of Sydney, Australia
A. John Hurst, Monash University, Australia
Alanah Kazlauskas, Australian Catholic University, Australia
Logan Muller, Unitec, New Zealand
Arnold Pears, Uppsala University, Sweden
Lynda Thomas, Aberystwyth University, Wales
Brian von Konsky, Curtin University of Technology, Australia

The Teaching of Novice Computer Programmers: Bringing the Scholarly-Research Approach to Australia

Tony Clear[†], Jenny Edwards[*], Raymond Lister[*], Beth Simon[‡], Errol Thompson[•] and Jacqueline Whalley[†]

[*]Faculty of Information Technology
University of Technology, Sydney
Australia
{jenny,raymond}@it.uts.edu.au

[†]School of Computing and Mathematical Sciences
Auckland University of Technology
New Zealand
{tony.clear,jacqueline.whalley}@aut.ac.nz

‡ Computer Science and Engineering Department
University of California, San Diego
La Jolla, CA USA 92093
bsimon@cs.ucsd.edu

[•]2 Haven Grove, Lower Hutt,
New Zealand
kiwiet@computer.org

Abstract

BRACElet is a multi-institutional multi-national research study of how novice programmers comprehend and write computer programs. This paper reviews the first action research cycle of the BRACElet project and, in the process, charts a path for the upcoming second cycle. The project remains close to educational practice, with much of the data being either data collected directly from exams sat by novices, or data from think-out-loud protocols where the task undertaken by a novice or an expert is modelled on an exam question. The first action research cycle analysed data in terms of the SOLO taxonomy. From think-aloud responses, the authors found that educators tended to manifest a SOLO relational response on small reading problems, whereas students tended to manifest a multistructural response. Furthermore, those students who manifested a relational response tended to do better overall in the exam than students who manifested a multistructural response. The second action research cycle will explore the relationship between the ability to read code and the ability to write code. Apart from reporting on the BRACElet project itself, this paper serves as an invitation for institutions and individuals to join the second action research cycle of the BRACElet project.

Keywords: Scholarship of teaching and learning, novice programmers, action research

1 Introduction

Across the western world, enrolments in IT degrees have decreased dramatically in recent years. Among the principal reasons for the downturn in student numbers are the perceived effects of the dot-com bust, media hype over outsourcing, the subsequent impressions formed by the parents of potential students, the poor teaching of computing in high schools and the 'nerdy' image of the profession.

All of the above reasons, but particularly the latter two, contribute to a further issue, the small and dwindling number of women entering IT courses. Edwards and Kay (2001) found that, despite years of special programmes for women in Australian universities, little has changed in twenty years. This is echoed in Camp's (1997) oft quoted work, *The Incredible Shrinking Pipeline*, and many other more recent publications.

Retention is also a major problem for IT education. While this is also due partly to the factors described above, first year teaching, especially the teaching of programming, is a critical factor. In a report from the international '*Grand Challenges in Computing Education*' conference, McGettrick et al. (2004) noted that educators cite failure in introductory programming courses and/or disenchantment with programming as major factors underlying the poor student retention in computing degree programmes.

1.1 Benchmarking Novice Programmers

It is well known, but often not discussed, that many undergraduates cannot program as well as their teachers would like. This is not a new phenomenon. Soloway et al. (1983) found that just 38% of computer programming students could write a simple program to calculate the average of a set of numbers, which is a task that most computing academics would regard as being well within the capabilities of a student who has completed a semester of programming. Perkins and Martin (1989) reported that students have fragile knowledge of basic programming concepts and a "shortfall in elementary problem-solving strategies". An entire volume of papers, called 'Studying the Novice Programmer', also documented the difficulties of learning to program (Soloway and Spohrer, 1989).

More recently, two ITiCSE working groups have benchmarked novice programming ability across several institutions and countries.

First, in 2001, the 'McCracken' working group assessed the programming ability of an international student

population from several universities (McCracken et al., 2001). The students were tested on a common set of program-writing problems and the majority of students performed far more poorly than expected.

Then, in 2004, the 'Leeds' working group (Lister et al., 2004) studied the code-reading skills of novice programmers. Data was collected from 615 students, spread across 12 institutions in 7 countries. The students were asked to answer several multiple-choice questions about short pieces of code. Again the majority of students did not perform well and approximately 25% of the students appeared to be performing at a level consistent with guessing.

1.2 The Nature of Expertise

There are many psychological studies of the differences between novices and experts in various professional, scientific, and artistic disciplines (Chi et al., 1988; Ericsson & Smith, 1991). Clearly, experts know more than novices, but psychological research indicates that experts also organize that knowledge into more sophisticated and flexible forms. This is apparent in the classic studies of chess players (Chase & Simon, 1973). When asked to memorize board positions of several chess pieces, novices tended to remember the position of each piece in isolation, whereas experts organized the information at a more abstract level, the attacking and defensive combinations.

Similar studies in computing show that expert programmers form abstract representations based upon what the code does whereas novices form concrete representations based on the code itself (Adelson, 1984; Corritore & Wiedenbeck, 1991; Soloway & Iyengar, 1986; Soloway & Spohrer, 1989; Wiedenbeck, Fix & Scholtz, 1993).

2 Research/Scholarship versus Folk Pedagogy

Our philosophy is that problems in teaching should be approached as research problems. In today's university, of course, this is not usually the accepted practice, to the considerable cost of our discipline.

While academics are very aware of events beyond their own institution that relate to their research interests, few academics are aware of teaching issues beyond their own institution. Few computing academics are aware of the above literature, both the literature of the psychologists and the literature that benchmarks novice programmers. Instead, computing academics tend to speak as if the difficulty many students face with programming is a new phenomenon – and furthermore an issue that is due to factors attributable and correctable entirely within their own institution (e.g. the quality of the lecturing, the choice of textbook, and the choice of language).

There is a considerable contrast between the research lives and the teaching lives of most computer science academics. In their research lives, they focus upon evidence and are part of a community that reaches beyond their own university. They read literature, attend conferences, carry out experiments, and publish, in a repeating cycle, with the individuals of a research community building upon one another's work. In contrast, the teaching lives of most computing academics are relatively isolated. As long ago as the 1960s, Ashby described the contradiction between the research and teaching lives of academics:

> *... all over the country ... scholars ... who would not make a decision about the shape of a leaf or the derivation of a word or the author of a manuscript without painstakingly assembling the evidence, make decisions about ... content of courses, and similar issues, based on dubious assumptions, scrappy data, and mere hunch.*
>
> Ashby (1963)

> *For many years I taught in universities. .... I marked thousands of examination scripts without examining what the scripts could teach me about my capacity as teacher and examiner.*
>
> Ashby (1985)

Echoing Ashby's observations, Bruner (1996) coined the term "*folk pedagogy*" to describe "*intuitive theories about how other minds work*". Bruner added that such folk pedagogies "*badly want some deconstructing if their implications are to be appreciated*". Similarly, Boyer (1990) argued that academic work could transcend the teaching versus research dialectic, if we saw academic work as comprising four equally important areas of scholarship, one of these areas being the "*scholarship of teaching*". However, as Fincher and Tenenberg (2007) recently noted:

> *Teaching remains rooted in practice, and not in its documentation. When teaching is documented, it is often in response to formal quality assurance requirements, or promotion procedures, not as part of a process of individual reflection and peer critique: these explicitly internal audiences ensure that such documentation, even when it exists, remains private. At the same time, the values and norms of educational institutions do not require or reward attribution in regard of teaching practice, rendering loss of provenance as almost inevitable. Such loss of provenance may in turn result in a loss of status of teachers amongst researchers sensitized to a research credit economy.*

Our philosophy is (1) that the process of acquiring programming expertise and the assessment of novice programmers should be approached as related research problems, and (2) that these problems, like all research problems, are most appropriately addressed by reading appropriate literature, conducting suitable experiments, and publishing the results of the experiments, so as to contribute to a global discourse on the teaching and learning of programming. This is the *scholarly* approach mentioned in the title of this paper.

3 The BRACElet Project First Iteration

The BRACElet project began in late 2004. It is committed to studying the problems of the novice programmer as a scholarly, research-led exercise. For its first year, the project organisation was ad hoc and opportunistic, but as BRACElet evolved, the collaborators began to understand the operation of the

project in terms of the principles of action research. As described by Carr & Kemmis (1986), action researchers "*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.*" (p. 28). In the case of the BRACElet project, the desired improvement of the real situation was an improvement in the exam assessment of novice programmers. A necessary support for this 'real' interest was a 'theoretical' interest in the Bloom and SOLO taxonomies, with the aim of (1) building theory about novices' acquisition of programming knowledge and (2) developing reasonable expectations about novice performance on test instruments. For a discussion of practice versus research in action research, see McKay & Marshall (2001).

Action research projects are often organized as a cycle (sometimes referred to as a spiral), with up to four stages occurring in each cycle: *Plan, Act, Observe*, and *Reflect*. The work and results described in the subsections below formed the first BRACElet action research cycle.

3.1 Reflection and Planning (Dec '04)

The first BRACElet event was a two-day workshop held at Auckland University of Technology in late 2004. The workshop began with participants reflecting upon the then recently completed Leeds Working Group (reviewed briefly in section 1.1 above).

The participants decided that, although the results of the Leeds working group were interesting, they were insufficiently underpinned by theory. The choice of the reading problems was not informed by a theoretical model – the problems were simply taken from past exam papers written by one of the project participants. Furthermore, the analysis of the data was not driven by any theory or model of how students should solve the questions they were given.

The BRACElet workshop participants decided that the revised version of Bloom's taxonomy (Anderson et al., 2001) provided a model for more systematically generating questions to put to students. The participants spent some time drafting questions with respect to that taxonomy. One of the styles of question drafted was the 'explain in plain English' question, as shown in Figure 1.

3.2 Acting (Jan-Jun '05)

In the semester after the workshop, project participants collaborated on preparing a common set of exam questions to include in their respective exams. Among these were 10 questions: 9 multiple-choice questions and one 'explain in plain English' question (the question shown in Figure 1). The 9 multiple-choice questions have been described in detail elsewhere (Whalley et al., 2006). These nine questions were consistent with the revised Bloom's taxonomy levels of *Understand* (3 questions), *Apply* (4 questions) and *Analyse* (2 questions).

The participants did not teach the same programming language, so questions were translated into the appropriate language for each institution – subsequent analysis has shown that programming language does not affect the results described below (Whalley, 2006).

In plain English, explain what the following segment of Java code does:

```
bool bValid = true;

for (int i = 0; i < iMAX-1; i++)
{
  if (iNumbers[i] > iNumbers[i+1])
    bValid = false;
}
```

Figure 1. An 'explain in plain English' question

SOLO category	Description of student's answer
Relational	A summary of the purpose of the code. For example, "*checks if the array is sorted*".
Multistructural	A line by line description of all the code. Some summarisation of individual statements may be included.
Unistructural	A description of one portion of the code (e.g. describes the *if* statement).
Prestructural	Shows substantial lack of knowledge of programming constructs or is unrelated to the question.
Blank	Question not answered.

Table 1: The SOLO Categories for the students' answers to the 'explain in plain English' question

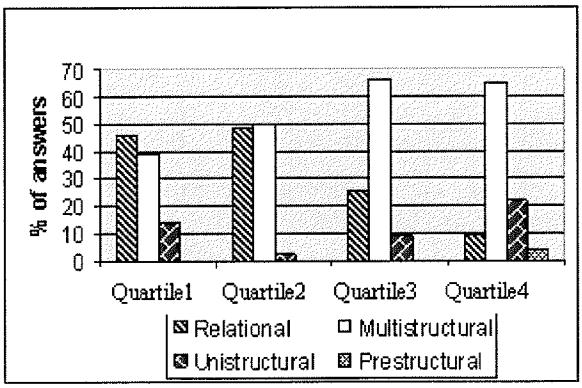

Figure 2. Performance on the 'explain in plain English' question, by performance quartile. (N=108). Quartile 1 is the top performing quartile. Figure reproduced from Lister et al., 2006.

3.3 Observing (Jul '05 - Jan '06)

The second BRACElet workshop was held in July 2005, as part of the NACCQ conference, shortly after the students had sat their exams. The workshop began the process of analysing the data from the exams – a process

that subsequently went on for about six months, with frequent email and VOIP exchanges between participants.

It was at this second workshop that the SOLO taxonomy (Biggs & Collis, 1982) was introduced to the project, as an agreed framework for categorizing the answers provided by students for the 'explain in plain English' question. The agreed categories are shown in Table 1.

The BRACElet participants have written several papers, describing in detail the analysis of the data collected from the exam papers (Whalley et al., 2006; Lister et al., 2006; Thompson et al., 2006; Whalley, 2006; Philpott, Robbins & Whalley, 2007; Thompson et al., 2008). In the remainder of this section, we shall discuss briefly only one result from those papers (Lister et al., 2006), as this is a result with direct implications for the design of the second iteration of the BRACElet project.

Figure 2 shows the distribution of SOLO responses for the 'explain in plain English' question, broken into four quartiles according to how well the students did on the nine multiple-choice questions. There are 27 students in each quartile, with quartile 1 being the top quartile. Approximately half of the students in the top two quartiles manifested a relational response to the 'explain in plain English' question. That is, approximately half of the students in the top two quartiles could 'explain in plain English' the purpose of the code (e.g. "checks to see if the array is sorted"). In contrast, multistructural responses dominated in the lower two quartiles. That is, students in the lower two quartiles tended to offer line-by-line descriptions of the code, without any indication of what computation the code performed.

Figure 2 may illustrate why many novice programmers struggle to write code. If we assume student responses to the 'explain in plain English' question are a reasonably consistent reflection of how the students reason about code, then it appears from Figure 2 that many of the weaker students do not naturally abstract from concrete code to ascertain the purpose of that code. Such an interpretation is consistent with the literature on expertise that was reviewed early in the paper.

3.4 Reflecting (Jan '06 - Jan '07)

Much of the reflection on BRACElet has been driven by conversations with people outside the project, often the audience at conference paper presentations. Such people have expressed three broad concerns with the BRACElet project, which we discuss in this section.

One of the concerns is that 'explain in plain English' is an ambiguous instruction. Audience members have argued that some students who might have been able to provide a relational answer (such as "checks to see if the array is sorted") instead thought that a line by line explanation was required. To some within the BRACElet project, this seems unlikely, for two reasons. First, why would fewer students in the top two quartiles misunderstand our instruction compared to students in the lower two quartiles? Second, one BRACElet participant has since started using 'explain in plain English' questions as part of her teaching throughout semester – making it clear that relational answers are preferred – and many of her students still do not provide a relational answer to such questions in the final exam. However, if there is to be a scholarly discourse on these exam questions, the BRACElet project should address this audience concern by devising an alternative experiment that (so the BRACElet participants expect) refutes this criticism.

The second concern rests on there being only one 'explain in plain English' question – if there were several such questions, would students reliably provide relational or multistructural answers?

The third concern is about the relationship between code reading and code writing. Some in the audiences raised the objection that writing programs and reading programs are disjoint intellectual activities. Such audience members argued that, even if students who provide relational answers are better equipped to answer the multiple-choice questions in our exam, such a result has no implications for code writing. To address such an audience concern, the BRACElet project needs to investigate the relationship (if any) between code reading and code writing.

4 The BRACElet Project Second Iteration

This section sketches our plan for the second iteration of the action research cycle. Furthermore, the commencement of a new iteration is the appropriate time to add new participants to the team, so this section is aimed principally at academics and institutions that are interested in joining the second iteration of BRACElet.

This second iteration of BRACElet will address the audience concerns described in the previous section, namely that (1) 'explain in plain English' is ambiguous, (2) 'explain in plain English' is an unreliable indicator of student ability to abstract from code, and (3) writing programs and reading programs are disjoint intellectual activities. The major stages for the second action research cycle of the BRACElet project are described in the following subsections.

4.1 Reflection and Planning (July - Dec '07)

Based on meetings, the existing BRACElet participants have developed the following broad plan framework for the second action research cycle. Exam papers written as part of the second iteration will contain at least three components:

1. A **SOLO component**, containing several questions (some of which may be 'explain in plain English' questions), where the responses to each question can be classified according to the SOLO taxonomy. One alternative to asking students to 'explain in plain English' is questions where students are provided code as an unnamed procedure/method, and are asked to nominate a suitably informative name for the procedure/method.
2. A **reading component**, similar to the BRACElet problem set from the first action research cycle.
3. A **writing component**, where students write a small piece of code. While the first action research cycle focused on improving the assessment for reading-type questions, the second cycle will also look at improving

the assessment of writing-type questions. Part of this process will be an examination of what makes a good writing question. If we can answer that, our investigation into any correlation between reading and writing code will be strengthened.

One of the BRACElet papers (Lister et al., 2006) ended with the following conjecture, which will be tested by this three-component framework:

> *If we identify two groups of students from the SOLO component – a group who reliably respond multistructurally and a group who reliably respond relationally – then the correlation of student performance on the reading component and the writing component will be weaker for the multistructural group than the relational group.*

The educational significance of this conjecture is as follows. In early university programming courses, many existing exam papers extensively reward concrete styles of reasoning – for example, by asking students to provide the value in a variable after a piece of code has finished executing, or by asking them to reproduce code that was studied in class during the semester – but students probably require a more abstract grasp of programming if they are to write their own computer programs successfully. If the above conjecture should be confirmed, then many programming teachers should rethink the validity of their assessment methods.

Each component of the study will also examine gender differences, both in the type of answers provided by male and female students, and in the attitudes of male and female students toward the different types of question. If there are gender differences, the computing discipline may be able to use this knowledge to address the poor recruitment and retention of women into IT courses (perhaps as part of a subsequent iteration of the BRACElet action research cycle). Furthermore, previous studies have shown that any educational approaches designed to attract more women into IT also attract more men, especially those from minority backgrounds.

4.2 Planning (Early 2008)

In early 2008, one-day workshops on BRACElet will be held in Brisbane, Sydney and Melbourne. These meetings will be open to all computing academics in those cities and surrounding regions. At these workshops, plans will be finalized for the second cycle.

4.3 Acting (First Half of '08)

Project participants will prepare exam papers that contain questions consistent with each of the three framework components outlined above (i.e. a SOLO component, a reading component, and a writing component). Participants will discuss drafts of their exam via email and VOIP. Participants will also complete local ethics clearance processes.

To assist participants in developing their exams, there will be a second one-day workshop in each of Brisbane, Sydney and Melbourne, with attendance restricted to those intending to incorporate the BRACElet framework into their exam papers. These workshops will occur about the middle of first semester 2008, when most participants will be preparing their exam papers.

4.4 Observing (mid-to-late '08)

In most institutions, students will sit an exam in June 2008, and the formal analysis of the BRACElet questions from those exams will commence in July. Taking the first action research cycle as an indication, the analysis phase of the second cycle is likely to take several months.

4.5 Reflecting (Oct '08 – Jan/Feb '09)

In the window Aug-Oct 2008, a two-day workshop will be held in Sydney, to reflect on the data analysis up to that point, and to have preliminary discussions on a third action research cycle. The workshop will form part of the International Computing Education Research Workshop.

4.6 Reflecting and Planning (Jan/Feb '09)

A BRACElet workshop will be held at the ACSW/ACE conference in 2009, in Wellington, New Zealand. A particular emphasis will be on engaging new BRACElet participants for a third action research cycle.

5 Conclusion

This paper reviews the first action research cycle of the BRACElet action research project and offers an invitation for interested parties to join the second cycle.

While the current BRACElet participants are confident that the second cycle will confirm and extend the findings of the first cycle, whether that actually happens is of secondary importance. Irrespective of whether the findings of the first cycle are confirmed or are found wanting, the participation of Australian academics in this study will educate them in the scholarly, evidence-based approach to teaching. Such an approach is an essential part of the restoration of computing as a popular undergraduate discipline.

Acknowledgements

The second action research cycle of the BRACElet project is funded by an ACM SIGCSE Special Projects Grant and an Associate Fellowship to Edwards and Lister from 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.

Anderson, L. W., Krathwohl, D. R., Airasian, P. W., Cruikshank, K. A., Mayer, R. E., Pintrich, P. R., Raths, J. & Wittrock, M. C. (Eds.) (2001) *A taxonomy for learning and teaching and assessing: A revision of Bloom's taxonomy of educational objectives*, Addison Wesley Longman.

Ashby, E (1963), *Introduction: Decision making in the academic world.* In Halmos, P (ed), Sociological Studies in British University Education, University of Keele. pp. 5-13.

Ashby, E. (1985), Preface to I. Brewer, Learning more and Teaching less, Guildford: Society for Research into Higher Education & NFER-Nelson, p. 4.

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

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

Camp, T. (2002) The incredible shrinking pipeline. *SIGCSE Bulletin,* 34, 2 (Jun. 2002), 129-134.

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.) *The nature of expertise.* Hillsdale, NJ, Lawrence Erlbaum Associates, 1988.

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

Edwards, J. and Kay, J. (2001) A Sorry Tale - the Participation of Women in Australian Computing Education, *Journal of Research and Practice in Information Technology*, Vol. 33, No. 4 329-335

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

Fincher, S. and Tenenberg, J. (2007) Warren's Question. Proceedings of the Third International Computing Education Research Workshop, Atlanta, GA, USA. September 15-16, 2007. http://www.cc.gatech.edu/conferences/icer2007/papers/1.pdf

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., (2004) A Multi-National Study of Reading and Tracing Skills in Novice Programmers, SIGCSE Bulletin, Volume 36, Issue 4 (December), pp. 119-150. http://portal.acm.org/citation.cfm?id=1041624.1041673&coll=&dl=&CFID=15151515&CFTOKEN=6184618

Lister, R., Simon, B., Thompson, E., Whalley, J. L., and Prasad, C. (2006). *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.

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.

McGettrick, A, Boyle, R, Ibbett, R, Lloyd, J, Lovegrove, L, and Mander, K. (2005) Grand Challenges in Computing: Education – A Summary. *The Computer Journal.* The British Computer Society. Vol. 48, No. 1. pp 42-48.

McKay, J., & Marshall, P. (2001). The dual imperatives of action research. *Information Technology and People, 14*(1), 46-59.

Perkins, D. N. and Martin, F., (1986), *Fragile Knowledge and Neglected Strategies in Novice Programmers.* In E. Soloway and S. Iyengar (Eds), Empirical Studies of Programmers, Ablex, Inc., Norwood, NJ, 213-229.

Philpott, A., Robbins, P. and Whalley, J. (2007) *Assessing the Steps on the Road to Relational Thinking.* 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. Mann, S. and Bridgeman, N. (Eds). ISSN 1176-8053.

Soloway, E. and Ehrlich, K, Bonar, J., and Greenspan, J. (1983) *What do novices know about programming*? In Shneiderman, B. and Badre, A. (Eds), Directions in Human-Computer Interactions, 27–53. Ablex, Inc., Norwood, NJ.

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

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

Thompson, E., Whalley, J., Lister, R., Simon, B. (2006) *Code Classification as a Learning and Assessment Exercise for Novice Programmers.* Proceedings of the 19th Annual Conference of the National Advisory Committee on Computing Qualifications, NACCQ, Wellington, New Zealand, July 7-10. pp. 291-298. http://bitweb.tekotago.ac.nz/staticdata/allpapers/2006/papers/291.pdf

Thompson, E., Luxton-Reilly, A., Whalley, J., Hu, M., & Robbins, P. (2008). *Bloom's taxonomy for CS assessment.* Proceedings of the Tenth Australasian Computing Education Conference (ACE2008), Wollongong, Australia, January 2008.

Whalley, J, Lister, R, Thompson, E, Clear, T, Robbins, P, Prasad, C (2006) *An Australasian Study of Reading and Comprehension Skills in Novice Programmers, using the Bloom and SOLO Taxonomies.* Australian Computer Science Communications 52: 243-252. http://crpit.com/confpapers/CRPITV52Whalley.pdf

Whalley, J. (2006) *CSEd Research Instrument Design: the Localisation Problem.* Proceedings of the 19th Annual Conference of the National Advisory Committee on Computing Qualifications, NACCQ, Wellington, New Zealand, July 7-10. pp 307-312. http://bitweb.tekotago.ac.nz/staticdata/allpapers/2006/papers/307.pdf

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., 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.