



Fostering Computing Education in NZ

Bulletin of Applied Computing and Information Technology

Technical Report T1:

07:01

2009/2010,
Dec/Jan



Report on the Eighth BRACElet Workshop:

BRACElet Technical Report 01/08 AUT University, Auckland

Tony Clear, Anne Philpott, Phil Robbins,

AUT University, New Zealand
tony.clear@aut.ac.nz

Simon,
University of Newcastle, Australia

Clear, T., Philpott, A., Robbins, P. & Simon. (2009/2010, Dec/Jan). Report on the Eighth BRACElet Workshop: BRACElet Technical Report 01/08 AUT University, Auckland. *Bulletin of Applied Computing and Information Technology* Vol. 7, Issue 1. ISSN 1176-4120. Retrieved December 7, 2009 from http://www.naccq.ac.nz/bacit/0701/2009Clear_BRACElet_Report.htm

1. Introduction

This paper reports on the activities of the Eighth BRACElet workshop held 4 July 2008 concurrent with the NACCQ Conference at AUT University. The BRACElet project is a longitudinal multi-institutional multinational investigation into the code reading, code comprehension and code writing skills of novice programmers.

2. Background - Workshops to Date

- The workshop reported on in this paper follows on from seven prior BRACElet workshops. A brief outline of the outcomes (paraphrased in part from Whalley & Robbins (2007)) follows.
- The inaugural BRACElet workshop was held in December 2004. During this workshop an instrument (a set of multiple choice and short answer questions) was established. Participants discussed the key ethical considerations and developed tools they would need to proceed with the study at their own institutions.
- During the second workshop held at the 18th Annual NACCQ conference in 2005, participants analysed and evaluated the results from the pilot studies. The results were employed to further hone the research instrument. The Bloom and SOLO taxonomies were used to try to gain a better understanding of the levels of difficulty of the MCQs (multiple

choice questions) and short-answer questions. The participants went away with a toolkit that allowed them to undertake a fully-fledged study at their respective institutions. The results from this work were disseminated at ACE 2006 (Whalley et al., 2006) and further papers were authored over the next few months by subgroups.

- A third workshop took place at AUT in late March 2006 and at this workshop we developed a prototype 'common framework' (Lister, Whalley, & Clear, 2006) that allows researchers to compare and contrast studies undertaken within BRACElet, but that also gives them the flexibility to tailor research to their particular interests.
- A fourth workshop was held at the 19th Annual NACCQ conference in Wellington 7th July 2006. The main purpose of the fourth workshop was to review the common framework, which had been initiated at the third workshop, and to continue developing it so that it would be useful for the next phase of the project, an investigation into the writing skills of novice programmers.
- A fifth workshop was held at the 20th Annual NACCQ conference in Nelson July 8th 2007. Goals were 1: to have some questions that fit the common framework in the coming semester's exam (or an equivalent), then to analyse, discuss, write and publish. Goal 2: Writing up the workshop for BACIT, for instance a critical evaluation of the questions talked about. Goal 3: Writing a paper for ACE2009. Anyone who had submitted an ethics application and had contributed draft exam questions in time for that deadline would be a co-author.
- A sixth workshop was held at AUT University 13-14 December 2007, sponsored by ACM & SIGCSE with nine institutions represented (7 New Zealand and 2 overseas). Raymond Lister, Beth Simon, and Errol Thompson each delivered a keynote presentation. Goals of the workshop were to analyse contributed data to develop new research instruments that would allow the evaluation of novice programmers' program-writing skills and enable comparisons to be made with their program reading skills. This toolkit was to be subsequently implemented at the participants' institutions in Australasia, the data analysed and joint publication(s) produced.
- The seventh workshop was held at Wollongong on 25th January at the ACE 2008 conference, with some 15 different institutions (mostly Australasian but one Swedish) present. This was the focal meeting for gaining Australian input related to the Carrick Institute Joint Associate Fellowship of Raymond Lister and Professor Jenny Edwards. An action research cycle was initiated at this meeting with the goal that this BRACElet iteration would: consolidate the "explain in plain English question"; generate new questions; examine any gender effects; examine any differences between international and local students (will have to be careful to differentiate between local "native speakers" and local ESL); relate answers to SOLO levels; examine differences between Undergraduate and Postgraduate; help develop an ideal exam paper.

3. The Goals of the NACCQ 2008 BRACElet Workshop

The purpose of this eighth workshop was to analyse assessment data from novice programmers contributed by participating institutions, thus furthering the inquiries into how novice programmers comprehend and write computer programs. Recognised educational frameworks of cognitive sophistication such as SOLO (Biggs & Collis, 1982) and BLOOM (Anderson et al., 2001) were to be applied as adapted for the programming domain. Analysis activities would include assessment of consistency of SOLO ratings using recognised statistical techniques. It was hoped that the outcome of this active workshop would include plans for data collection and further analysis relevant to each institution.

4. Participants

The workshop was attended by nine participants from one overseas institution and three institutions within New Zealand (Table 1). Other AUT participants attended the presentation sessions.

Table 1. Sixth workshop participants

Institution	Participant
Auckland University of Technology (AUT)	Tony Clear
	Anne Philpott
	Gordon Grimsey
	Phil Robbins
Manukau Institute of Technology	Mike Lopez
	John Peppiatt
	Kirsten Marais
University of Newcastle	Simon
Southern Institute of Technology	Ken Sutton

5. Preliminary Agenda for BRACElet workshops

Appendix A below gives the initial agenda for the workshops, which were initially planned as two half-day sessions, but with most of the attendees present from the outset, effectively became one session, continuing throughout the day.

6. Progress of the workshop

The workshop began with a brief review of the SOLO analytical taxonomy as adapted in prior BRACElet work for application to program reading tasks. An interpretation of the taxonomy was outlined from the NACCQ conference paper by Clear et al. (2008), to update workshop participants on the classification approach. These categories are tabulated in table 2. The workshop then moved into a practical session in which the participants broke into small groups in order to analyse a corpus of data brought to the workshop, and classify student responses according to the SOLO categories given in table 2. Sets of questions and student responses from assessments and examinations from AUT, SIT and University of Newcastle were available as data for analysis, covering different classes and question types, both reading and writing tasks.

Table 2. SOLO Categories Derived in Clear et al., (2008)

SOLO category	Description
Relational [R]	Provides a summary of what the code does in terms of the code's purpose. (The "forest")
Relational Error [RE]	Provides a summary of what the code does in terms of the code's purpose, but with some minor error.
Multistructural [M]	A line by line description is provided of all the code. (The individual "trees").
Multistructural Omission [MO]	A line by line description is provided for most of the code, but with some detail omitted.
Multistructural Error [ME]	A line by line description is provided for most of the code, but with some minor errors.
Unistructural [U]	Provides a description for one portion of the code.
Prestructural [P]	Substantially lacks knowledge of programming constructs or is unrelated to the question

The groups worked in pairs or threes to jointly conduct their analysis.

6.1. Analysis groups

Three groups worked independently to analyse and classify the contributed data applying the given SOLO categories.

The **first group** worked on introductory writing tasks, using the AUT-supplied data from a set of very concrete exercises and responses (n=37) assigned to Programming Fundamentals students - a CS0-type course for a cohort who had been struggling with the standard CS1.

The **second group** worked on a more intermediate set of paired reading and writing questions and responses (n=29) from a University of Newcastle examination at the CS1 level.

The **third group** worked on a more advanced level (CS3) set of student responses (n= 12) to a pathfinder algorithm tracing assignment.

6.2. Analysis groups - findings

The **first group** found themselves trying to apply descriptions that related to code reading tasks to a code writing task, and felt that although the SOLO categories were valid, new descriptions were needed. It was also felt that, unlike with the code reading tasks, the code writing tasks were quite restricted in that even a correct answer could not be said to demonstrate relational thinking.

The students had been given a Book class with two attribute variables and one accessor method. Question 1 was "write an accessor method, getName, to return the name of the book". A correct answer required two lines of code (a signature and a return statement) plus braces. Question 2 was similar requiring a mutator method.

Question 3 was "complete the method `isAvailable` so that it returns false if the book is on loan but true otherwise". A skeleton method had been provided with a local Boolean variable that had been initialised to true and returned. There was an instance variable, `onLoan`, which was true when someone had borrowed the book. Students were expected to write a simple if statement (returning `!onLoan` was possible, but not expected from these students).

The group felt that these questions were of the Bloom type "change in representation" as the requirements were almost pseudocode.

The group suggested that the following list (table 3) might be suitable SOLO descriptions:

Table 3. Suggested SOLO Categories Applicable to Program Writing Tasks

SOLO Category	Description for a code writing task
Relational	Ability to correctly write code to solve a problem which has not been specified to the extent that the problem represents pseudocode for the solution. Example - write a class to represent a library book
Multistructural	Ability to correctly write code requiring a small number of statements e.g. an if statement. Ability to write a correct multi-line solution based on pseudocode. Example - complete a method so that it returns false if the book is on loan but true otherwise
Multistructural error	As above, but where a single error prevents the code from working correctly, e.g. a wrong variable has been used.
Unistructural	Ability to correctly complete a single small piece of code. Example - write a line of code to increment the variable count
Prestructural	Unable to write correct code.

Based on this, most of the group's responses were either multistructural (the code worked) or prestructural (the code either did not compile or did not work at all).

The **second group** produced a set of categorisations for questions 23 and 25f from the dataset, which traversed the full set of available SOLO classifications. A brief demonstration of the application of interrater reliability on the dataset for question 23 using SPSS, and categorical data analysis with Kendall's W, indicated a very high degree of agreement [.994] between the pair of raters (see table 4 below), however the rating was conducted jointly by consensus, so high agreement would be expected.

Table 4. Newcastle Question 23 SOLO Classification Test Statistics

N	2
Kendall's W(a)	.994
Chi-Square	55.687
df	28

|A symp. Sig. .001|

a: Kendall's Coefficient of Concordance

Action Point: This dataset could now be independently rated by a further set of raters, which might prove illuminating.

The **third group** worked on a more advanced level (CS3) set of student responses (n= 12) to a pathfinder algorithm tracing assignment. These responses required some careful analysis before beginning the SOLO classification. Classifying the task itself took some time. On referring to the question types of the "Common Framework" in table 5, the group concluded that it was a new type of task not represented to date.

Table 5. Common Framework component IV: the Assessment Framework (ex. Whalley & Robbins, 2007)

Question type	task	Description or example
Fixed code	reading	Requires the student to hand execute (trace through) some code and select, from a set of provided answers, the correct outcome or result (Lister et al., 2004).
Skeleton code	reading	Requires the selection of the correct code, from a set of provided answers, which completes the provided "skeleton" code (Lister et al., 2004).
Change in logic	reading	Questions where the student is given a code fragment and the solution is a code fragment that should give the same result but the logic of the algorithm has been altered (or reversed)
Change in representation	reading	Questions where the student is given a code fragment and is asked to identify the same program in an alternative representation or vice versa (Whalley et al., 2006)
	writing	Question type drawn upon Soloway's extract questions (1986). Given an algorithm in pseudo code or plain English translate this logic into <i>valid</i> code in language X.
Code purpose (explain in plain English)	reading	Given a code segment, explain the purpose of that piece of code. (Whalley et al., 2006, Lister et al. 2006)
Classification	reading	Classify a number of code segments that are very similar by identifying the similarities and differences in structure and/or purpose
Code refactoring	reading	Given the original piece of code and several options (answers) for refactoring the code, critique the answers or given a piece of code identify "smells" in the code by highlighting sections of the code (see Smells to Refactoring - [Kerievsky], 2006).
	writing	Refactor the provided code (in the form of a class) by decomposing it into methods.
Parson's Puzzles (see Parsons & Haden, 2006)	Bridging the reading/writing divide?	Given the purpose of a snippet of code and the code snippet in which the lines of code are mixed up (out of order), the code

		statements must be rearranged into the correct order for the code to run successfully.
Code intent	reading	From a test case or series of test cases, determine the intent, by explanation or answering directed questions or by writing the actual class(es) for which this test specifies the functional intent
	writing	

The task required students to simulate how a pathfinder algorithm operated after observing the operation of the game software, and then being given a question with set positions marked on a grid (representing a board including occupied squares a starting square and a target square). In addition students were given a printout of the code and a sample dataset, and asked to depict that simulation of finding the shortest path to the target. In some respects it equated to a more sophisticated "fixed code - reading" type of question, and in other respects it was a "change in representation - reading" type of task. Mapping the task against the revised Bloom taxonomy (Anderson et al., 2001), we concluded that the task was probably beyond the "apply" level and better reflected an "analyse" task. We defined it as a new type of question, a "simulation task".

While initially thinking that the task as assigned was innately "multistructural", on conducting the SOLO rating the group encountered one relational response wherein the student had not only traced the algorithm, but critiqued it by making observations about the advantages and disadvantages of the algorithm's design. In addition they noted that the task of devising a suitable representation to depict the response was in fact a second task, which they assessed as being at the more advanced revised Bloom's level of "create", and perhaps innately demanding a relational level of response. Had a predefined response structure been specified (e.g. scaffolding by providing some form of empty diagrammatic or tabular input framework to be filled in etc.) then a more multi-structural response level may have been achievable? Responses were further categorised by whether students had represented the "data" elements of the simulation and/or the "process" elements

6.2.1. Reflections on the ease of SOLO Classification

Several implications arise from this work of the three groups in relation to SOLO classification.

- 1) The purpose of SOLO classification was unclear to some participants, and its value as a benchmarking and differentiating tool with an underlying theoretical basis needed to be explained.
- 2) The guidelines provided in table 2 from the paper by Clear et al. (2008) did not appear to guarantee consistency in classification, and it appeared that a workshop approach to develop a common understanding was required (e.g. for RE what is a 'minor error' that retains a Relational coding, rather than a relational "fudge" which should more fairly be coded as unistructural or prestructural?).

The "descriptions" in table 2 above addressed reading tasks only, not writing tasks, and needed expansion.

The existing "descriptions" also needed expansion to better phrase the concepts, if rating is to be conducted based only on those descriptions - but given the comments just made in 2) is that a valid premise?

Action: expand descriptions (Tony Clear)

3) The implicit value system, with its hierarchy of cognitive sophistication from prestructural upwards, proved a distractor for raters. The applicability of different "natural" SOLO ratings for different types of question did not appear immediately obvious or legitimate. This also led to arguments about the "ordinality" of the scale, especially with RE and MO set within the implied hierarchy. The scale was evidently "categorical", however, which satisfied the form of analysis in the paper. In future work it could be statistically tested for a tendency towards ordinality, at least at the major points of the scale. However, the intermediate points (RE, & ME) raised some questions, and should perhaps be excluded from any ordinal scale if ordinality was to be a confirmed property.

4) The rating exercises did suggest that "MO" was not readily distinguishable as a category from "ME" and should be culled from the scale, making it more parsimonious. So the workshop has also resulted in a useful refinement of the reading scale, depicted below in table 6, but still with the limitation of its code "reading" focus in the descriptions.

Table 6. Modified SOLO Categories from those in Clear et al., (2008)

SOLO category	Description
Relational [R]	Provides a summary of what the code does in terms of the code's purpose (the "forest").
Relational Error [RE]	Provides a summary of what the code does in terms of the code's purpose, but with some minor error.
Multistructural [M]	A line by line description is provided of all the code (the individual "trees").
Multistructural Error [ME]	A line by line description is provided for most of the code, but with some <i>minor errors</i> .
Unistructural [U]	Provides a description for one portion of the code.
Prestructural [P]	Substantially lacks knowledge of programming constructs or is unrelated to the question.

5) Finally, table 7 depicts the combined reading and writing SOLO scale, developed as a result of the workshop. This modified classification scheme now needs to be tested in the field. Notably an 'RE'category description for writing is absent, and it remains to be seen whether this category will become evident from further work with SOLO and 'writing' tasks.

Table 7. Modified SOLO Categories to Reflect both Program Reading and Writing Tasks

SOLO category	Description of Category in the Programming domain	
	Reading Tasks	Writing Tasks

Relational [R]	Provides a summary of what the code does in terms of the code's purpose (the "forest").	Ability to correctly write code to solve a problem which has not been specified to the extent that the problem represents pseudocode for the solution. Example - write a class to represent a library book
Relational Error [RE]	Provides a summary of what the code does in terms of the code's purpose, but with some minor error.	
Multistructural [M]	A line by line description is provided of all the code (the individual "trees").	Ability to correctly write code requiring a small number of statements e.g. an if statement. Ability to write a correct multi-line solution based on pseudocode. Example - complete a method so that it returns false if the book is on loan but true otherwise
Multistructural Error [ME]	A line by line description is provided for most of the code, but with some <i>minor errors</i> .	As above, but where a single error prevents the code from working correctly, e.g. a wrong variable has been used.
Unistructural [U]	Provides a description for one portion of the code.	Ability to correctly complete a single small piece of code. Example - write a line of code to increment the variable count
Prestructural [P]	Substantially lacks knowledge of programming constructs or is unrelated to the question.	Unable to write correct code.

7. Future work & publication ideas

Arising from the workshop a set of specific follow-up actions, publication targets and authorship teams were then set, as a way of carrying the work forward to tangible outcomes.

Specific **actions** agreed were:

- 1) arrange access to BRACElet wiki at AUT for participants - Simon, Ken, & Mike (Tony)
- 2) this to include access to the original question sets from the early papers (Tony)

- 3) Mike to be given email list of BRACElet project members (Tony)
- 4) MIT students portfolio data to be included in next set of analysis, ethics approvals to be arranged (Mike Lopez)

Proposed publications with due dates and teams are outlined in table 8 below. This invitation to contribute naturally also extends to Raymond Lister as one of the principals of BRACElet, who was unable to make this workshop due to ITiCSE commitments.

Table 8. Publication Targets

Papers	Ideas
ACE 2009 (Sep 07)	The BRACElet project [tentatively correlation between reading and writing questions from the exam - using Newcastle, SIT and AUT contributed data]
	(Simon, Ken, Phil & Mike)
SIGCSE 2009 (Aug 29)	Algorithm simulation analysis
	(Anne, Tony, Jacqui)
This BRACElet Workshop Report [T/R]	Tony, Anne, Phil & others who may contribute

References

- Anderson, L. W., Krathwohl, D. R., Airasian, P. W., Cruikshank, K. A., Mayer, R. E., Pintrich, P. R., Raths, J., Wittrock, M. C. (Ed.). (2001). *A Taxonomy for Learning and Teaching and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*, : Addison Wesley Longman Inc.
- Biggs, J. B., & Collis, K. F. (1982). *Evaluating the quality of learning: The SOLO taxonomy (Structure of the Observed Learning Outcome)* . New York: Academic Press.
- Clear, T., Whalley, J., Lister, R., Carbone, A., Hu, M., Sheard, J., et al. (2008). Reliably Classifying Novice Programmer Exam Results using the SOLO Taxonomy. In S. Mann & M. Lopez (Eds.), *21st Annual NACCQ Conference* (Vol. 1, pp. 23-30). Auckland, New Zealand: NACCQ.
- Kerievsky, J. (2006). *Smell to Refactoring Cheat Sheet* . Retrieved 02 Aug 2008, from http://weblogs.java.net/blog/wwake/archive/2006/05/smell_to_refact.html
- Lister R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, E., Sanders, K., Seppälä, O., Simon, B., and Thomas, L. (2004). A multi-national study of reading and tracing skills in novice programmers. *SIGCSE Bulletin* 36:4, 119-150.
- Lister, R., Whalley, J. & Clear, T. (2006). For Discussion: A Framework for a Meta-Project on Students Programmers (BRACElet Technical Report No. 0106). Auckland: Auckland University of Technology.
- Parsons, D. and Haden, P. (2006). Parson's programming puzzles: a fun and effective learning tool for first programming courses. *Eighth Australasian Computing Education Conference (ACE2006)*, Hobart, Australia, 157-163.
- Soloway, E. (1986). Learning to program = learning to construct mechanisms and explanations. *Communications of the ACM* 29:9, 850-858.
- Whalley, J., Clear, T., & Lister, R. (2007). The Many Ways of the BRACElet Project. *Bulletin of Applied Computing and IT*. Retrieved June 3, 2007 from http://www.naccq.co.nz/bacit/0501/2007Whalley_BRACELET_Ways.htm, 5 (1).

Whalley, J. L., Lister, R., Thompson, E., Clear, T., Robbins, P., Kumar, P. K. A., & Prasad, C. (2006). An Australasian Study of Reading and Comprehension Skills in Novice Programmers, using the Bloom and SOLO Taxonomies. *Proceedings of the Eighth Australasian Computing Education Conference (ACE2006)*, Hobart, Australia CRPIT, 52 : 243-252.

Whalley, J. & Robbins, P. (2007, Jun), Report on the fourth BRACElet workshop. *Bulletin of Applied Computing and Information Technology* Vol. 5, Issue 1. ISSN 1176-4120. Retrieved December 17, 2007 from http://www.naccq.co.nz/bacit/0501/2007Whalley_BRACELET_Workshop.htm

03 Aug 2008

Appendix A



NACCQ BRACElet Workshops - 2 Half Days - Morning and Afternoon

"A Multi Institutional Study of Program Reading and Writing Skills in Novice Programming Students".

Purpose

These workshops are open both to current participants in the Bracelet project and those with an interest in analysing how novice programmers acquire skills in reading and writing programs.

The workshops will be run as active working sessions with the aim of producing a tangible output, and informing the next phase of the study for each participating institution. It is hoped that the outcome of the workshop will include plans for data collection and further analysis relevant to each institution. It is intended that the results of the workshop may be published in the form of a Bracelet technical report.

The workshops have been designed in two parts, to enable participants to either participate for the full day, or attend other half day workshops if they wish.

The Bracelet study, initiated in 2004, has been inquiring into how novice programmers comprehend and write computer programs. Participants will analyse assessment data from novice programmers contributed by participating institutions. Recognised educational frameworks of cognitive sophistication such as SOLO and BLOOM will be applied as adapted for the programming domain. Analysis activities will include assessment of consistency of ratings using recognised statistical techniques.

Preparatory work for the workshop

For the morning session participants are requested to bring with them any data collected from their institutions' examinations or other assessment instruments. Ideally this data will include two pre-identified code reading and code writing questions with answers, deemed by the contributors to be at a comparable cognitive level.

This data may be sent to the coordinators beforehand, or attendees will bring them along as resource materials for discussion and review at the

workshop.

For the afternoon session participants should be able to work with the data contributed in the morning session.

- **Facilitators:** Tony Clear, Anne Philpott, Phil Robbins

NACCQ BRACElet Workshop Agenda (AUT University 4 July 2008)

****MORNING SESSION****

11:00-11:40 Intro & welcome. Brief presentation outlining SOLO taxonomy from forthcoming NACCQ 2008 paper, and reviewing contributed data available for analysis. Select one or two questions and work through answers together assigning SOLO categories to responses to demo working of process.

11:40-12:30 Break into two groups and conduct SOLO analyses on assigned questions and datasets. Capture results into spreadsheets.

One group will work with Phil's data from the programming fundamentals group.

The other group will include Anne's "simulation task" assessment data for more experienced students, which will involve more exploratory analysis of the intersection between reading and writing code. The artefacts are richer with a focus on the design of algorithms. We hope to capture useful results nonetheless. Questions here will first involve task identification, then categorisation. Working with this set of data is expected to continue after lunch.

12:30-1:00 Lunch

1:00-1:30 General, audience led discussion about the outcomes, issues identified, category boundary definitions and exemplars. Review of participants' contributed questions and their relationship to categories identified.

1:30-2:00 Continue with working on Anne's data in two groups.

2:00-2:20 Discussion relating to insights arising.

End with a quick (10 minute?) preview of the remainder of the day, in case anyone is still wondering whether to stay or go to another workshop.

****AFTERNOON SESSION****

2:30-4:30 Brief recap on morning's work for new participants. Introduction to statistical methods for assessing reliability of multiple raters.

Continue to work in groups, but individually conduct analysis of interrater reliability [IRR] from assigned question (explain in plain English from Phil), using SPSS and "Kendall's W".

May also need to have one group replicate analysis of Anne's data, in parallel, prior to assessing IRR.

3:30-4:00 Afternoon Tea

4:30-6:00 General, audience led discussion about the findings, highlighting problem areas for further analysis, and a plan ahead for publication and future work at each institution.

In groups review questions with low interrater reliability, and discuss problem questions. Where possible, develop agreed rubrics to accurately classify responses, and highlight boundaries between classifications.

Discuss plans for extending BRACElet in each institution.

Publications Discussion

A revision of the BRACElet "compact" (e.g. how you get to be a co- author on the next study).

Arrange for jointly authored publication(s) arising from the workshop, e.g. Technical Report, BACIT special edition/section, targeted conferences e.g. ACE, Koli Calling, SIGCSE?, ITICSE

Copyright © 2009 Tony Clear , Anne Philpott , Phil Robbins ,Simon

The author(s) assign to NACCQ and educational non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced. The author(s) also grant a non-exclusive licence to NACCQ to publish this document in full on the World Wide Web (prime sites and mirrors) and in printed form within the Bulletin of Applied Computing and Information Technology. Authors retain their individual intellectual property rights.

Copyright © 2009. NACCQ, Michael Verhaart, Krassie Petrova and Nick Wallingford (Eds.). An Open Access Journal, DOAJ # 11764120.
(✓ zotero)