Transforming Learning of Programming: A Mentoring Project

Daryl D'Souza[1] **Margaret Hamilton**[1] **James Harland**[1] **Peter Muir**[2]
Charles Thevathayan[1] **Cecily Walker**[1]

[1] School of Computer Science & Information Technology
RMIT University,
PO Box 2476V, Melbourne, Victoria 3001,
Email: {daryl.dsouza, margaret.hamilton, james.harland, charles.thevathayan, cecily.walker}@rmit.edu.au

[2] SET Portfolio Office
RMIT University,
PO Box 2467V, Melbourne, Victoria 3001,
Email: peter.muir@rmit.edu.au

Abstract

Programming is central to Computer Science and cognate disciplines, and poses early-learning challenges in problem-solving and coding. Since the recent past the School of Computer Science & Information Technology (RMIT University) has provided a student mentoring service to assist novice student programmers with their programming, indeed, to build up their confidence in programming. The service has received favourable feedback from students and, as an interesting aside, has had the added benefit of increasing mentors' confidence and improving mentors' communication skills. Mentors volunteer their services under a University leadership initiative, and are not paid to assist students. In light of such success, we secured a University action-research teaching and learning grant, to investigate aspects of the service delivered to date. While mentoring has been shown to be helpful for novice student programmers to learn and improve their programming, less recognised, but of equal importance, is the value to mentors through the skills and experience they gain. This paper reports early findings of a dual-purpose research investigation into the mentoring service. The research project seeks to discover ways to improve the mentoring service for novice student programmers, as well as to enhance a range of qualities in mentors.

Keywords: Mentoring, Introductory Programming, Action Research

1 Introduction

Programming is central to Computer Science and cognate disciplines and forms a core activity in related courses, offered in Computer Science and Information Technology schools or departments. In this context, programming is a key survival skill for students and permeates almost all study and assessment activity in relevant courses. For this reason, commencing students enrol in programming subjects in the early stages of their study plans. Schools make every attempt to therefore teach programming well at this early stage, and typically allocate disproportionate resources to ensure successful, long-term outcomes, in terms of building students' skill levels and confidence in programming.

Despite such attempts and allocation of resources, programming continues to pose early-learning challenges for students with little or no programming background. At the individual level, these challenges often stem from an inability to link a range of activities with the practice of programming, such as problem analysis and problem-solving, through to coding and testing. Therefore, it is essential to build a strong foundation for students, in the formative years of programming.

At the school level, resource constraints present their own challenges to develop such a programming foundation. Increased resourcing is often countered by the need to group together diverse student cohorts, whose entry level scores are widely distributed. Moreover, students are enrolled in a range of undergraduate and postgraduate courses, have varying programming ability. Students may be articulating or transitioning from other courses, often undertaken at other universities or TAFE (Technical and Further Education) colleges, such that they join our degree courses in the second or third year. Furthermore, local and international students display different levels of confidence in their approach to seeking help from staff. Within our school we have run programming help desks using paid student tutors for several semesters, yet have found them to be underutilised and too general to be of great use to students.

From our student progress data, it is clear that where students are struggling with programming, it affects most facets of their study, for example: their progress through their study program, their study habits, their confidence, and their time management. Electronic assignment submissions allow us to see who has not submitted assignments. Students who have not submitted assignments often emerge as those who struggle with understanding the tasks and in general the subject. Some students are afraid to ask for help or, due to work commitments, are unable to access fixed consultation times of lecturers and head tutors. The lack of access could be one reason why many students feel alienated. Electronic discussion boards are useful but are frequented by a core of students, who are more confident and willing to publicly ask their subject-related queries and submit other relevant contributions. The volume of discussion board content often grows rapidly, with anecdotal evidence indicating that this sometimes leads to disillusionment with the facility, a phenomenon especially noticeable among the weaker students, who might otherwise benefit most from such a forum.

Having a mentoring program is one way to re-establish universities as communities of scholars rather than as mere venues for formal classes. This opportunity was met a few years ago when our school was approached to trial a mentoring service which had been successfully implemented in the Faculty of Business. The trial service proved to be successful for both mentors and mentees, and have since continued the service. In light of some notable early successes in the mentoring service, we secured a University action-research teaching and learning grant for semester 2, 2007, to explore several aspects of the service delivered over the previous three semesters. Our objectives are to understand the key success features of the mentoring program and to investigate the effects of mentoring from two perspectives: mentors and students (who access the mentors). The first is to examine aspects of students' experiences of using the mentoring service. For example: why do they attend, and what do they get out of it? The second angle is to explore mentors' experiences: recognising that mentors are student volunteers, and several opt to mentor for more than one semester, what is it that they gain from being a mentor, and does it offer them chances to strengthen specific skills? And, what are their reasons for volunteering?

While mentoring has been shown to be helpful for novice student programmers to learn and improve their programming, in part due to students' preference for seeking help from peers, less recognised, but of equal importance, is the value to mentors through the skills and experience they gain. This paper reports early findings of a research investigation into the mentoring service, with a dual purpose. The research project seeks to discover ways to improve the mentoring service for novice student programmers, as well as to enhance a range of qualities in mentors. To the best of our knowledge no previous work has simultaneously explored both mentor and student experiences in an action-research context, for novice student programmers. Our aims are to establish a better mentoring service for this student cohort and, in the medium to long term, to establish a far-reaching mentoring practice within the school, for all programming subjects and beyond.

The remainder of this paper is structured as follows. In Section 2, we present some definitions, including a brief discussion about the difference between the roles of mentor and tutor, a brief history of the mentoring service, and a review of related work. In Section 3, we present our research methodology: its action-research basis and the research framework of our project. In Section 4, we document some outcomes and provide some discussion thereof. Finally, we present the conclusion and intentions for future work in Section 5.

2 Background

2.1 Definitions and context

A *mentor* is "a wise or trusted advisor or guide"[1] and has its its origins in Greek mythology, when King Odysseus placed his old age friend, Mentor, in charge of his son, when Odysseus left to fight in the Trojan War.[2] Accordingly, a mentor engages in *mentoring*, a term used to refer to counselling or training of new employees or students by an experienced individual.

In our mentoring service *mentors* are currently enrolled students who are deemed highly capable of advising novice student programmers. They are considered as being expert student programmers. *Mentees* are beginning or novice programming students with little or no programming background enrolled in one of two targeted programming subjects. One of these is a first-year subject with Java as a vehicle for instruction. The other is a second-year subject that teaches C at a moderately complex level; students are expected to have prior knowledge of programming. Mentees are encouraged to see mentors during the week at times when mentors are available. The mentoring service has a set timetable, offering access at various times during the part of the semester when the service is made available. Mentors are scheduled for a two-hour session each week, for 8 weeks, beginning in week 4 of semester. They are always scheduled in pairs, partly to ensure that sessions are always serviced, and also for mutual support between mentors.

2.2 Mentors versus tutors

The question often arises: What distinguishes mentoring from tutoring? We provide our response in the context of our own mentoring service, and hope that the distinctions presented are widely applicable.

First and foremost, in our mentoring service tutors are paid while mentors volunteer their services, under the University leadership program available to students, known as the LEAD (Learn, Engage, Aspire, Develop)[3] program. The program invites students to participate in activities that allow them to improve their communication and leadership skills. Notwithstanding the mechanism by which mentors initially become involved, more fundamentally, we note the following distinctions between mentoring/mentors and tutoring/tutors.

- Mentors provide personalised attention to students. Tutors operate in a classroom context and address groups of students, preferably in discussion mode.
- The attention offered by mentors usually manifests itself in the form of guidance and is student-driven. The student (or mentee) approaches the mentor with problems for which the mentor is unprepared, other than being the domain expert or facilitator. Tutors (and tutorials, where students receive attention) present planned teaching and learning objectives, which by implication limit the discussion to the content of tutorial sheets.
- Mentors address arbitrary problems presented by students, within the context of the mentoring service. Unlike a tutorial, which is aimed at enhancing understanding of scheduled topic coverage, mentors are expected to be able to address relevant issues with no temporal or topical limitations.
- By virtue of its flexibility, mentoring does not place a time limit per se on mentee-mentor discussions, offering yet another flexibility dimension not offered by tutoring and tutors.
- Mentoring and tutoring are complementary. A mentor offers individuals a chance to get their own needs met, whereas tutorials offer a forum for the exchange of views with others.
- In broad terms mentors (and mentoring) offer a differentiated teaching and learning service, with a view to providing a non-prescriptive yet facilitated learning approach. Tutoring, on the other

[1] The Collins English Dictionary, Australian Edition, Edited by G. A. Wilkes, 2nd edition, 2001.

[2] en.wikipedia.org/wiki/Mentor, www.aare.edu.au/02pap/jan02265.htm

[3] www.rmit.edu.au/browse;ID=qrkzw11526gb1

hand, is planned and script-driven, in its teaching and learning objectives. It invites and encourages two-way communication between tutor and students, albeit within the context of the given script (the tutorial sheet).

In summary, a paid tutor is obliged to offer a structured and planned opportunity for students to engage in discussion about a specific, recently-discussed lecture topic. A (volunteer) mentor offers a consulting service to students (mentees), which is unplanned and student-driven.

2.3 A brief history

In this paper we describe an ongoing investigation of a mentoring project being conducted in our school. At the time of writing the project is still underway and expected to conclude after the end of semester 2, 2007. Analysis of data collected to date is expected to be completed by late November, 2007. The project is being funded by a University action research teaching and learning grant, based on a successful, recent past trial of a mentoring service offered to students enrolled in two targeted subjects, in which the focus is teaching novice student programmers.

Two years ago the School was approached by the Science, Engineering & Technology Faculty to trial a mentoring service which had been previously implemented with some success in the Business Faculty. The mentoring service in Business was applied at the Faculty level, which was possible because of the common curriculum framework prevalent in undergraduate Business courses, particularly first year. However, with its discipline diversity this was not possible within our Faculty, so a trial within a single school in our Faculty seemed the more appropriate option.

Our school was chosen due to the interest of two staff, one from within the School and one from the Faculty, in offering mentoring or peer-tutoring to students in their first year of study. The Faculty staff member was able to offer support for a project within a school. Our school was considered an ideal choice in which to trial the service because of its cohort diversity, multiplicity of courses, and a known problem of high attrition rates in students' first programming subject. The School is also unique in that it employs teaching and learning advisors, both of whom have higher qualifications in Education, and who potentially bring a complementary perspective to such a project, over and above the ideas of the other academic (teaching) staff.

In the initial trial during semester 1, 2006, a single programming subject was targeted, and mentors were selected from among latter year students with excellent results in the same subject. As mentioned earlier, mentors volunteered under the University LEAD program, and selected mentors were timetabled across the working week for students (mentees) to access them. Limited mentor access was also made available after 5pm, for part-time (mature age) mentees to be able to seek mentoring help. In the third semester of operation, the mentoring service was extended to include a second programming course, also for novice programmers. Mentors were selected in the same way, and the number of mentors was increased to cater for the larger potential source of mentees. While offering help in more than one programming language was found to be attractive to mentees, increasing the number of mentors meant that timetabling and organising training became more complex. Thus, for the current running of the program (its 4th semester), numbers were reduced, as it was noted that the majority of mentors had strong skills in both of the programming languages supported by the mentoring service. The practicality of having mentors assist students in both languages was appealing to the mentors as it broadened the skills they could use, and it also simplified timetabling and allowed more flexibility in times mentees could seek help.

2.4 Related work

The work of Miller & Kay (2002), discusses the introduction of mentors across all of their first year computer science courses. They note that "The fact that the majority of mentors spend the majority of their time on programming and Java issues shows that the more social issues (transitional, personal and administrative issues) were not perceived as the focus for mentors or their students." A similar scheme trialled in another school in our University generated a mentor dropout rate of 50%. In another study of mentoring in computer science, Miliszewska & Tan (2007) introduced a system where two mentors provide support for one hour every day at the same time. They report a 10% improvement in the pass rate, and an improvement in grade average and student satisfaction for their course.

Several researchers have investigated issues relating to the difficulty of learning to program (Simon et al. 2006, McCracken et al. 2001). Jenkins (2002) covers many factors, such as "Few students find learning to program easy". He considers the questions of aptitude, cognitive factors such as learning styles, motivation, handling multiple skills, multiple processes, learning the language, educational novelty, interest, reputation and image of being a nerd, and pace at which materials are covered. The most common comment he claims to hear from students is that programming is "boring and difficult".

Assessment is another key concern to many researchers. Fekete et al. (2000) recommend that students reflect about their programming, and they describe a range of ways to encourage students to reflect on the state of their knowledge, and the process by which students acquire it. Web-based arrangements for students to practice assessing material using specific criteria for marking are provided and marks are allocated in assessment for reflective writing. Lister et al. (2003) recommend a grading philosophy aimed at encouraging each student to realise their potential. They recommended grading be done by assigning a grade to a student according to how that student meets explicit clear criteria, "irrespective of the resultant grade distribution" according to Bloom's taxonomy.

Some authors have considered how students learn (de Raadt et al. 2005) and how to measure motivation, possibly via the Biggs study process questionnaire (Biggs 2001). The Biggs survey relates to the Biggs Revised Two-Factor Study Process Questionnaire (R-SPQ-2F) (Biggs 2001). Henceforth, we will refer to this Questionnaire as the *B*iggs survey. The Biggs survey is designed to ascertain learning approaches (surface, deep and achieving) of university students. Student profiling techniques combined with adaptive learning technologies to detect learning patterns and traits in students have been recommended by Traynor (2004) to record how students learn. They argue that the empirical evidence can be used to reason about the development of a cognitive model of learning programming. However, despite these advantages many modern attempts to teach computer science electronically seem to fail. They state that: "Unlike corporate training where the material is reasonably simple to digest, or linguistics where there is a large body of literature concerned with its education, computer science education is neither simple nor well documented."

All of the above strategies, implementations and measures have had varied successes by improving the learning environment for CS&IT students. The best way forward appears to be in establishing support for programming students early on in their learning experience. Stamouli et al. (2004) set up a programming support centre, but concluded that even after it had operated for a year with considerable publicity, many students were still unaware of it. However, for those students who did access it, the benefits were well worth their while. We consider that while many factors contribute to students having difficulty in learning to program, the supporting human factor is one largely overlooked and difficult to measure.

3 Research context

3.1 Research methodology

We have employed the action research methodology to our study. Action research relies on planning, applying some intervention known as the action, reflecting on the changes this action has brought about and planning further similar cycles. Specifically, we employ the proven methodology of Checkland, which has been applied in a range of educational and organisational settings (Checkland 1981, Checkland et al. 1990, 1998).

As action research, the project will follow an iterative cycle of steps that involve planning, action, review, further planning and so on. This project is conceived as one cycle (June–November) that builds on the experience gained from the trial of the mentoring program since the beginning of 2006 and will inform the subsequent development of the program in 2008 and beyond.

A feature of action research is the importance of collaboration or participation of all key stakeholders in all steps of the research cycle. In the context of this project the following participant groups will be involved.

1. Mentees
2. Mentors
3. Project investigators
4. Relevant teaching staff—lecturers and tutors in selected programming subjects

Access to the mentoring service is optional, so teaching staff involved in the targeted subjects, Programming 1 and Programming Techniques, have been asked to continually publicise the mentoring service to their students, via all available mechanisms, including online discussion boards, during consultations, and periodically in classes (lectures, tutorials and laboratories). Indeed, staff are actively encouraged to identify students who appear to be struggling, and to refer them to the teaching and learning advisors as well as to the the mentoring service. These referrals are aimed at increasing access to the service, especially for students who are likely to be most at risk of not succeeding. Teachers of the subjects involved in the mentoring project, however, will not participate as investigators, and we will merely seek their collaboration to encourage students to access the mentoring service.

In this study, the action is placing the mentors in appropriate classrooms for interaction with voluntary students. We reflect on why students volunteer to be participating mentors; why students approach mentors; and what impact the mentoring has on the overall learning of the students involved in the process.

A range of mechanisms will be used to gather data for analysis, documentation and future directions. These are listed in Table 1. The focus group

Activity/technique	Stakeholders
Desk research (historical data)	Investigators
Semi-structured focus group interviews	Mentors/Mentees
Biggs survey	Mentees
E-journal reflective feedback	Mentors
Reflective journals	Investigators

Table 1: Activities and participants in mentoring, action-research project

interviews were conducted separately for mentors and mentees. That is, a different set of questions is formulated for each cohort. The Biggs survey is distributed to mentees during the early part of semester and closer to the end of the semester. It represents one component of the quantitative data we collected for research purposes. The other component is collected via a software system which allows mentors to register mentee visits. Such data will be used in conjunction with the Biggs data, to ascertain the impact of mentoring on students' learning habits, hence the conduct of two Biggs' surveys.

The reflective journals were established in 2006, in earlier trials of the service, and has proved to be very successful and informative. Journal entries are submitted by mentors via their own personal accounts and accessible only to them and to the investigators. Reflections entered are general statements about effectiveness, and do not identify mentees or specifics about mentoring session with mentees.

We hope that our action research methodology will allow us to formally assess the following two aspects of the mentoring program: how it can best address the learning needs of novice programming students, and its value to the volunteer mentors. The significance of the project is to build/improve an effective peer mentoring program. It is anticipated that the project will have benefits for all the key groups involved in the project, specifically,

Mentees

- Independent learning - self diagnosis of difficulties in programming
- Building confidence in programming
- Opportunity to improve overall study habits and learning approaches

Mentors

- Leadership and leadership training
- Problem diagnosis
- Peer tutoring skills

School Staff (project investigators; teaching staff—lecturers, tutors)

- Improvements to student learning outcomes for mentees
- Provision of timely and relevant support for students
- Leadership and peer tutoring opportunities for mentors
- Validation and refinement of the peer mentoring for application in other CS&IT subjects (and programs)

3.2 Description of project

Mentor training involved the following approach. Training for each semester was divided into three sessions: an introductory session of approximately one hour, and two two-hour sessions, one run during the semester, and the other at the end of the semester. Training focused on two key aspects; the first covered learning and teaching principles, particularly those related to peer-tutoring and working one-on-one or with small groups. Topics included learning styles, building rapport and questioning. The second aspect of training focused on critical issues faced by novice programmers. To assist this, a scenario guide was developed.

The scenario guide provided strategies for mentors to deal with different situations, posed by mentees. Each scenario represents a diagnostic and the guide lists suggested ways to deal with the mentee. The underlying theme in all advice or guidance is to be proactive yet non-prescriptive. Mentors are also trained to ensure that the mentee leaves with a sense of confidence and control over their learning objectives, in the context of the query presented during their visit.

During the first training session, a discussion was also held about a code of practice for mentors to follow. Colleague experiences of running mentoring programs elsewhere indicated that it is advisable for mentors to create a brief set of guidelines to use to enable them to set boundaries both on a personal level and to ensure that they do not become a 24-hour helpdesk. For example, mentors are not expected to give out their mobile phone numbers, and are not required to provide their email addresses to mentees. At the outset, mentors are unclear about why this is needed, however they come to appreciate that some guidelines are useful after participating in mentoring.

To participate in the the LEAD program, mentors are required to reflect on their experience, and a staff member is required to monitor and comment on their work. We requested that our mentors write a brief reflective piece after each mentoring session, addressing the following questions and points:

- When did I feel most effective as a mentor?
- When did I feel least effective as a mentor?
- I have the following comments to make on the mentoring session/program in general.

From semester 1, 2007, the support of the University's centralised IT services was garnered to include the mentor program in our online teaching repository. A site was created for mentoring, and mentors were each given an e-journal where they could enter their reflections electronically, and where staff could read and comment on their work regularly, rather than waiting for hard copies to be handed in. This has proved a valuable tool, as the input from mentors has led to several refinements of the service each semester. The e-journals have also allowed us to pick up critical issues as they arise, such as learning issues detected within a specific student cohort, or issues related to the use of the mentor lab. Feedback to individuals is possible via the e-journals, whereas group communication with mentors can be done through email.

4 Outcomes

4.1 Discussion

One of the most challenging elements of running the mentoring service has been to find ways to encourage students to use the service, that is, to be mentees. While there is clear evidence that mentoring is useful for novice programmers (Miller & Kay 2002, Miliszewska & Tan 2007), our experience was that the students who most needed help were those least likely to access the service. It was pleasing to discover that many students who used the service returned again and again. However, it has been necessary each semester to rethink ways to encourage students to use the mentoring service as often as possible and in a way which will also help them to be better organised. In the first semester of mentoring, particularly as it was new to the School both for staff and for students, there were many sessions which were not attended at all. In the second semester, we introduced mentors by getting a group of five volunteers to attend one lecture. While staff had announced the mentoring service many times, actually seeing the faces of fellow students was considered a way to have them recognised as peers. Indeed, feedback indicated that students might be inclined to visit a mentor if they knew that there was someone there from their own cultural background, or who would possibly speak their first language.

Further efforts were made to increase attendance to the mentoring service in first semester 2007, when tutors and lab assistants were asked to personally recommend mentoring to any student who was having difficulties. And for the current semester, we have integrated an early intervention exercise with mentoring, where teaching and learning advisors track students who have not submitted an early piece of work in one of the core programming courses, and one element of the assistance given to these students is to recommended that they seek help from mentors. They are provided with a summary of how mentoring can assist them, and a personal copy of the mentoring timetable. While use of the service has increased slightly each semester, this aspect of mentoring remains a challenge. It is hoped that data from the current project will offer ideas to increase the use of mentoring services.

4.2 Mentor feedback and discussion

Obviously the mentors are essential to the success of this intervention. We surveyed them at the end of their semester and asked the following six questions. We were surprised by the number of students who volunteered to be mentors and hence the first question on our evaluation sheet was the following one.

a) How did you envisage your role before you commenced as a mentor? Did the reality match this concept? Why? Why not?

Most of the responses centred around the fact that the mentors were prepared to share their immense knowledge, but the students required very specific help, such as the solution to the problem with their code.

> "The mentor will have more time to explain something but after I became a mentor, I just like to point out some error and have them to correct the error. It is hard to explain and make them understand. It is easier to solve their problem (point out the error without the source of the error) without made them understand completely about their problem."

> "I imagined I'd be helping people understand the broader concepts of programming. In reality I spent much more time teaching students to debug their code more effectively."

> "Before I began, I saw my role as a mentor as an opportunity to assist others in developing their

skills in an area I am strong in. However, I found that people came in with very specific questions/problems and (I) wasn't as easily able to 'generalise' the problems as I'd hoped."

"One on one, same student each week. reality was multiple students and different students each week."

Many of the mentor responses are truly indicative of the lecturer's feelings of disappointment, when they realise they may not have reached the students:

"I expected to actively participate and share my knowledge with other students. In reality, I had only one student to work with and it was quite hard to understand her problem."

"I thought I could help people to understand concepts. This worked a bit, but generally the students were more interested in getting me to solve their problems for them."

However, there were more successes than disappointments as the responses to the following, second question indicate:

b) Now that you have acted as a mentor:

- **When have you felt most effective? Why?**

"I felt most effective when I had explained or went over something with a mentee, and then seeing them later be able to overcome that (and similar) challenge(s) by themselves."

"When the student (mentee) doesn't understand concepts. Explaining concepts is easier to convey than debugging code."

"I felt most effective when I was able to help the java kids understand why/why not something was working with their code. Also watching them learn from what I'd said, and applying it."

"When I was able to help student grasp a concept that they had previously not understood."

"I can give advice to the student. Most important. I can guide them how to solve the problem by themself"

"I felt most effective when a student came back on more than one occasion because he liked my help so much."

"When a student faced a problem similar to one I had faced so I could relate to their problem."

As shown above, the mentors had the same sense of satisfaction when the student has understood the concept. However, also, as shown below, they share the same frustrations as lecturers when students stay away from their classes and do not understand what is being explained.

- **When have you felt least effective? Why?**

"Basically the opposite of the above; ie if the same person came to me with the same or a similar problem later, because it made be feel like I had failed in teaching them the skills they need."

"When no-one turned up, I felt fairly uneffective."

"I felt least effective when I could see that the student was just there to get answers and when they left the room they were no better off than when they came in"

"When students didn't understand what I was trying to show them."

c) What issues/problems/ logistical difficulties (if any) have arisen?

There were some issues with the room being noisy, or small, but several mentors felt problems were due to their inability to communicate effectively:

"Speaking different languages (not enough vocabulary) "

"I got a little problem explaining the problem due to its complicated nature and that I'm not used to explaining things. Anyway, overall it was fine."

"Most students ask for help WITHOUT knowing exactly what it is they need help for. Some students expect assignment solutions. They tend to dismiss any relevant information mentors give them."

"Roster more mentors for expected high-use times, such as the last day before an assignment deadline."

The answers to the following question really demonstrated the true understanding the mentors had of their mentees, and their close identification with them.

d) What helpful feedback can you give on improving the program's operation?

"Provide lunch after meeting"

"Maybe run more sessions near the end of the week as many more students showed up then in line with assignment due dates."

"I felt that the students I was helping were doing quite well and just had small difficulties. I would have liked it if some more struggling students came along"

"The name 'mentor' sounds a bit odd. I would have preferred to call myself a helper (and did from time to time)"

"Lecturer should encourage student come to the mentor, and encourage them to ask good questions so we can share the idea."

e) What, if anything, have you gained from your participation in the program?

"Meet new friends from same sector; gain knowledge about context and noise "

"A greater awareness of my own strengths and weaknesses"

"More experience helping other people. I personally feel the ability to help some one is a good skill to have, and this mentoring program helps sharpen those skills and improve self-esteem."

"I would like to think I have improved my communication skills and are able to explain things more clearly."

"Got to know other mentor(s). More confident with teaching strangers. Improved teaching and communication techniques."

"... an interesting insight into basic teaching skills and also the problems commonly faced by those starting Java and programming in general"

"Confidence. Improved problem solving. Made friends. Developed listening and explanation skills. Experimented with different teaching styles"

"Be slow, think things through, ask questions instead of giving answers."

The variety of responses above are excellent morale-boosting responses for the rewards of being a teacher. The strength of these responses has encouraged us to continue with this program of mentoring, and to undertake this study, with a view to recommending it's much wider application.

4.3 Mentee feedback and discussion

Feedback was also solicited from all students in the final survey, whether or not they used the mentoring service. While Biggs survey data has yet to be analysed we asked students three questions about mentoring, as part of the final Biggs survey. Responses were sought from students who visited mentors as well as others, who did not. The questions (in bold font), student responses (in quotes) and our comments appear below.

1. **If you used the mentoring service indicate whether you found it useful and, if so, in what ways.**

   "I had so many 'ahhhh!!' moments that I eventually lost count. "

   The student is referring to that feeling that one gets when a program finally works, or a bug is identified at long last. They go on to describe the mentor-mentee exchange that led to this moment of satisfaction. The mentor first convinces the student that their program is doing what *they* (the student) has directed the program to do.

   "The mentor that assisted me was extremely helpful in guiding me through the thought process behind the code and also with his explanations as to why I was getting the errors which is instrumental in my basic overall comprehension of programming. "

   "I would show him my code and tell him just how defiant my program was and how it "isn't listening to me, I tell it what to do and it JUST WON'T LISTEN," to which he not only explained that my code was, in fact, doing EXACTLY what I was telling it to do, but he also walked me through the reasoning behind my errors."

   "He would then have me attempt the code again after a discussion of pseudocode and when my code wouldn't work, he again explained why until I worked out for myself what was actually going wrong. "

   "Often times when working on my code, I would start off on the reasonably right track, however when I'd start to get the 'red squiggly lines' in Eclipse, I'd get extremely creative in my ways of attempting to rid my program of them. So in the end, I wouldn't really understand why my program was working but took delight in seeing that my squiggly lines were gone. "

   The student conveys the realisation that the brainstorming with the mentor has been of value, and wishes that they had had the mentor-mentee dialogue much earlier. They also hint at the service being expanded beyond Programming 1 (this student happened to be enrolled in Programming 1).

   "That's not to say that my creative programming is a thing of the past, but I now see the value of taking the time to seek assistance for better comprehension of what I'm doing. My only regret is that I haven't taken advantage of this service back in week 4. "

   "I am really grateful for the mentoring service which you have organised. All the mentors I have seen gave me a great assistant in my programming. I often make some silly mistake such as forgotten to input the curly bracket in the for loop, Prompting user to input, declare variable which is not going to be used in the program."

   "The mentor often advises me to write down on paper what the outcome would be before I do any coding. Yet I often write the program without thinking through carefully and hoping the program will run perfectly. "

   "It is amazing to see that each mentor will have their own ways of writing the program. I have learnt a lot from them. I really appreciate to all mentor who have devote their time and knowledge to help the others who need help. I wish I can be as clever as they are."

   "All mentors have been really great for devoting their times and patient to explain what the code will do for each program. I just like to express my gratitute towards all mentors who have given me a helping mind in the coding. "

   "Peer support is very useful as they (the peers) belong to same age group, you can seek their help anytime you like."

   Peer assistance aside, mentoring was seen as providing a range of benefits, emphasising its student-driven focus. The service was described variably as: being useful, suggesting that it allowed students to touch base on minor matters; providing valuable individual attention; and representing a handy walkthrough service!

   "Yes it is useful, get help to solve problems."

   "I found it quite useful, as the mentors could usually help out."

   "The one-on-one help was a pleasant change from the labs, where all we ever do is demo."

   "Yes, as a second eye to what I can't see."

   "Yes it does clear the doubt that is in my head with regards on errors when compiling programs."

2. **If you didn't use the mentoring service, please tell us why not.**

   The responses contain important information that will help us improve the service. Some issues for consideration include better notification of and about the service, as well as the need to emphasise that better guidance from mentors is far better than seeking help from friends, who are less likely to be able to use the guided approach used by mentors.

"Some of my friends who were good at programming helps me when I need help so I did not go to the mentoring service."

"Didn't need it."

"I think I can solve problems by myself or asking friends."

"Because I get enough help from tutorial and lab."

"Did not know the time. Needed help with assignments, not sure if they could help."

"Other commitments/travel."

"Times didn't suit subjects."

"Because the mentor doesn't really help you to really understand what you are doing which cause you have problem in the next phase's study."

The last response suggests an expectation on the part of the student (mentee) for more than just guidance in the form of a strategy. There is a need for the mentor to better equip or empower the mentee towards their goals; at the same time, the mentee has not fully understood this rationale behind mentoring—the the mentor is not there to provide answers but to provide food for thought for that "next phase".

3. **Please provide any other comments you have about mentoring.**

Both negative and positive responses were received. The following negative response is an expression of frustration of having to wait for attention—frustrating for the student, but a sign that service popularity resulted at times in long waiting queues, suggesting perhaps the need for bookable sessions.

"If the mentors put down a time to help and don't they shouldn't bother making us wait for them to never show up."

The following comment is not surprising. While we trained mentors to assist with assignment questions in a guided fashion, clearly some students did not value such guidance when assignment-related queries were posed. This is as much about mentor training as it is about student understanding of the service, and suggests the need for associated improvements. Mentors need to at least provide some hope to mentees when parting with mentees who pose assignment-related queries. At the same time, mentees need to understand guidance and facilitation are the objectives of mentoring, not spoon-feeding.

"Should have mentoring for assignments beyond questions."

Finally, two interesting remarks follow. One is a request for an online mentor, something that left us wondering and reflecting on the possibilities. An online chat service, perhaps, with an electronic whiteboard mechanism? The second remark is an endorsement from a student who was not enrolled in a subject for which the service was targeted.

"It would be great if mentoring was available online too!"

"It rocks as I am in it as well for a different subject though."

Further anecdotal (verbal) evidence suggested that the service was well received and students recommended that it be expanded. Indeed, the mentee focus group was attended by students who used the service, yet who were not enrolled in subjects for which mentoring was available! To sum up, the key issues of mentoring that were cited as essential, included the following (not an exhaustive list).

- Continuity in terms of mentor availability.
- Personalised attention.
- The guided approach to teaching.
- Access to mentors near a lab, yet isolated from the lab, so that one could migrate between lab and mentor at will.
- Consultation with peers.

5 Conclusions and future work

This paper has highlighted the need for a humanistic approach to enhance teaching and learning of programming in the context of novice (student) programmers. Mentoring has been shown to conceptualise the learning of programming in a university context, with benefits that complement other teaching contexts, such as tutorials. Our particular contribution is the use of action research to explore enhancements in the teaching by mentors, who afforded individual attention to mentees, in their learning of programming. The designer of the C++ programming language, Bjarne Stroustrup, said (Stroustrup 1997): "Design and programming are human activities; forget that and all is lost." We have found this idea to also apply to the learning of design and programming. Without the human interaction, learning to program can be very lonely, isolating and depressing. The mentors bring another human element back into the activity. We have found this human element to be extremely important to the struggling novice programmers for many reasons. Firstly it shows them that others have struggled and succeeded, so it is possible to continue. Secondly the mentors are trying to speak to them in a language they understand, and so their advice is often clearer. The mentors are teaching from a position of already having passed the subject which the lecturer is delivering, but the lecturer is a further step removed from the students, and cannot always get their attention in the same way. Finally, the mentoring is personal, one on one, so the student can feel their needs are really being met.

Mentoring provides an opportunity for students to ask whatever questions are troubling them in relation to programming. As we discussed in section 2.2, mentoring differs from tutoring in this aspect, as tutors can have many students in their room, they cannot spend a long time explaining concepts in detail to individual students who may not have grasped something from two weeks ago. In this sense, tutoring and mentoring complement each other as teaching modes. Also, since the mentors are not part of the teaching team, they do not know the details of the assignments. Each mentee had to explain their individual problems, and it was this articulation of the problem which proved to be the most beneficial to both mentors and mentees. The pride of the student who found their red squiggly lines removed from Eclipse, was equally matched by the mentor's satisfaction in reaching out to a student and helping them understand the magic. Also, the surprise of one student in seeing that each mentor had a different way of tackling his problem, was palpable.

From the point of view of the mentors, they are being introduced to the practice of teaching in small doses, and on the whole most thoroughly enjoyed the experience. As discussed in the outcomes, we were overjoyed by the enthusiasm of the mentors, and are trying to "bottle it up" to keep for other subjects and students as well as for these introductory subjects in future semesters. Following mentors' suggestions we are planning to set up a wiki with a view to allowing future mentors the opportunity to learn from the previous ones, and so build up a supportive network. The wiki will allow them to communicate in real time, and share ideas and code fragments for assisting mentees.

We can conclude from the responses of students who have used the mentoring service that mentoring can be particularly beneficial to international students. For example, we know anecdotally that international students may be far more comfortable approaching another student due to their reticence to ask for help, particularly from a lecturer or tutor, and particularly when they are new. The cultural adjustment required of them in a new country often compounded by the extra workload required of them to catch up with the other students. In the past many such students have either dropped out of the programming courses or failed these courses badly. The mentors have been able to build up their confidence by encouraging them, and by filling any gaps in knowledge required for the early assignments. It is noteworthy that approximately 25%–30% of the mentors themselves were international students, for whom this scheme provided an excellent opportunity to improve their communication skills. They have been only too happy to volunteer their services, and are proud to contribute to helping others learn.

One of the significant but indirect benefits of this study has come from the feedback cycle created by the mentoring system. We have noticed that students are much more willing to convey their conceptual difficulties to their mentors whom they face and speak to on a one to one basis. Many mentors have in turn conveyed the common difficulties faced by their mentees to the teaching staff through informal feedback and emails. The teaching staff are then able to address some of these issues during classes, thus completing the feedback cycle well before the end of semester.

While substantial data analysis is yet to take place, in the spirit of action research, and with feedback already received, we reflected about future directions, following this first cycle in our mentoring exploration. Some pursuits will come in the form obvious improvements. These include the afore-mentioned use of a wiki to allow mentors to engage with each other about their experiences; an added bonus is that such documented experiences will feed back into improvements to the mentor scenario guide mentioned in Section 3.2. Other related improvements identified during this cycle, include: additional training needs to be incorporated in mentor training; early announcements to students of the mentoring service; encouragement and intervention schemes for students to see mentors as early as possible; the need for booked mentor times during peak periods; improvements to physical space requirements.

On a wider scale, we have also identified the following directions, to be incorporated in future action research cycles.

- The mentoring service would not have been possible without the LEAD initiative and the significant support provided by our teaching and learning advisors. We need to examine the impact of such contributions, if the service is to be transplanted in other contexts. How might it run without LEAD or teaching and learning advisors?
- An example of other contexts where the service may be applied is to make it available to students enrolled in *any* programming subject. As well, it may apply across the board, to all first year subjects. What additional requirements are necessary for such broadening of the service?
- Somewhat orthogonal to the previous application context, an interesting comparative study might be in the use of the mentoring service in a different discipline. Every discipline is likely to present its own "bottleneck area" in the sense of causing frustration among students over their understanding of foundation-level material, that is considered critical to succeeding in the entire course. One example is the discipline of Psychology where reportedly one of the discipline bottlenecks is the study of statistics. Can the mentoring service be customised to other disciplines?
- Another potential contribution of mentoring is in preventing or reducing plagiarism. The symptoms that lead to high attrition rates are often the same symptoms that lead to plagiarism. We believe that mentoring can, to some extent, alleviate plagiarism, by empowering students to take control of their own learning of programming.

In summary, we believe that future action research into mentoring can address these issues, and in several concurrent cycles, if carried out in cooperation with others (disciplines, schools, universities).

6 Acknowledgements

We acknowledge the contributions of others who have been directly involved in the project, including Jeanette Holkner, a project team member; Megan Kek, from the Science, Engineering & Technology Portfolio; Peter Tilmanis, who has provided a range of technical support activities, including the development of mentee visit registration system; and Shuhaida Shuhidan for her ongoing contribution as the project research associate.

We especially thank the students who volunteered their services as mentors and are grateful for their willingness to participate in the research project. We also thank all the students who actively sought help from mentors, and we are grateful to those students who participated in the focus group interviews and who completed the Biggs survey questionnaires.

Finally, we acknowledge the support of the University in terms of the LEAD project, the portfolio for funding the project, and the School of Computer Science & Information Technology.

References

Biggs, J., Kember, D. & Leung D.Y.P. (2001), The revised two-factor Study Process Questionnaire: R-SPQ-2F , British Journal of Educational Psychology Vol 71(1), pp. 133–149.

Checkland, P. & Holwell, S. (1998), Information, systems and information systems - making sense of the field , Chichester, John Wiley & Sons.

Checkland, P. & Scholes, J. (1990), Soft systems methodology in action , Chichester, John Wiley & Sons.

Checkland, P. (1981), Systems thinking, systems practice , Chichester, John Wiley & Sons.

de Raadt, M., Hamilton, M., Lister, R., Tutty, J., Baker, B., Box. I., Cutts, Q., Fincher, S., Hamer, J., Haden, P., Petre, M., Robbins, A., Simon, Sutton, K. & Tolhurst, D. (2005), Approaches to Learning in Computer Programming Students, and Their Effect on Srld, HERDSA 2005, pp. 407–414.

Fekete, A., Kay, J., Kingston, J. & Wimalarante, K. (2000), Supporting reflection in introductory computer science, *in* 'ACM SIGCSE Bulletin', Vol. 31(1), pp. 144–148.

Jenkins, T. (2002), On the Difficulty of Learning to Program *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences pp. 53–58.* Retrieved August, 2007 from `http://www.psy.gla.ac.uk/~steve/localed/jenkins.html`.

Lister, R. & Leaney, J. (2003), First Year Programming: Let All the Flowers Bloom *Proceedings of the 5th Australasian Computer Education Conference (ACE2003), Adelaide, Australia, pp. 221–230*

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y., Laxer, C., Thomas, L., Utting, I. & Willusz, T. (2001), A multi-national, multi-institutional study of assessment of programming skills of first year CS students *in ACM SIGCSE Bulletin*, Vol. 33(4), pp. 125–140.

Miliszewska, I. & Tan, G. (2007), Befriending Computer Programming: A Proposed Approach to Teaching Introductory Programming , Issues in Informing Science and Information Technology Vol 4, pp. 278289, 2007.

Miller, A. & Kay, J. (2002), A Mentor Program in CS1 , ITiCSE 2002, June 24-26, Aarhus, Denmark, 2002.

Simon, Fincher, S., Robbins, A., Baker, B., Box, I., Cutts, Q., de Raadt, M., Haden, P., Hamer, J., Hamilton, M., Lister, R., Petre, M., Tolhurst, D. & Tutty, J. (2006), Predictors of Success in a First Programming Course *in* Computing Education 2006, CRPIT, Vol. 52, pp. 189–196.

Stamouli, I., Doyle, E. & Huggard, M. (2004), Establishing structured support for programming students *in* 'Proceedings of the 34th ASEE/IEEE Frontiers in Education Conference, Savannah, GA. Retrieved August, 2007 from IEEE Xplore `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1408612`.

Stroustrup, B. (1997), The C++ Programming Language Third Edition Addison-Wesley Publishing Company, 1997.

Traynor, D. & Gibson, J.P. (2004), Implementing Cognitive Modelling in CS Education:Aligning theory and practice of learning to Program , IADIS International Conference CELDA 04, 15th - 17th December, 2004.