Research in Learning Technology
Vol. 23, 2015

A mixed-methods exploration of an environment for learning computer programming

Richard Mather*

Department of Computing, Buckinghamshire New University, High Wycombe, United Kingdom

(Received 6 January 2015; final version received 28 July 2015)

A mixed-methods approach is evaluated for exploring collaborative behaviour, acceptance and progress surrounding an interactive technology for learning computer programming. A review of literature reveals a compelling case for using mixed-methods approaches when evaluating technology-enhanced-learning environments. Here, ethnographic approaches used for the requirements engineering of computing systems are combined with questionnaire-based feedback and skill tests. These are applied to the 'Ceebot' animated 3D learning environment. Video analysis with workplace observation allowed detailed inspection of problem solving and tacit behaviours. Questionnaires and knowledge tests provided broad sample coverage with insights into subject understanding and overall response to the learning environment. Although relatively low scores in programming tests seemingly contradicted the perception that Ceebot had enhanced understanding of programming, this perception was nevertheless found to be correlated with greater test performance. Video analysis corroborated findings that the learning environment and Ceebot animations were engaging and encouraged constructive collaborative behaviours. Ethnographic observations clearly captured Ceebot's value in providing visual cues for problem-solving discussions and for progress through sharing discoveries. Notably, performance in tests was most highly correlated with greater programming practice ($p \leq 0.01$). It was apparent that although students had appropriated technology for collaborative working and benefitted from visual and tacit cues provided by Ceebot, they had not necessarily deeply learned the lessons intended. The key value of the 'mixed-methods' approach was that ethnographic observations captured the authenticity of learning behaviours, and thereby strengthened confidence in the interpretation of questionnaire and test findings.

Keywords: evaluation; mixed methods; video analysis; computer programming

Introduction

A raised profile for 'Computer Science' highlights challenges in learning to program

Primary and secondary school curricula in the United Kingdom are being revolutionised by the replacement of the ICT curriculum with computer science (Burns 2012; Department for Education 2013a, 2013b, 2013c). This is driven by government recognition of global shortages and gender imbalances in supplies of school leavers to higher education and in computer science graduates to industry. This important

*Email: Richard.Mather@bucks.ac.uk

Responsible Editor: Meg O'Reilly, Southern Cross University, Australia.

Citation: Research in Learning Technology 2015, **23**: 27179 - http://dx.doi.org/10.3402/rlt.v23.27179
(page number not for citation purpose)

transformation is clearly building momentum under the auspices of the grassroots organisation 'Computing At Schools' and with the support from key professional, industry and academic bodies, such as Microsoft Research and the British Computer Society (Brown *et al.* 2013).

As a result of transformations in computing education, approaches to teaching computer programming at all levels of education are the subject of special review. Regarding technologies for teaching programming, a great variety of platforms are now available. Examples used for pedagogic comparisons, such as reported by Fincher, Cooper, and Maloney, 2010, include Scratch (Resnick *et al.* 2009), BlueJ (Kölling *et al.* 2003), Alice (Cooper, Dann, and Pausch 2000), Greenfoot (Henriksen and Kölling 2004) and Lego Mindstorms (Barnes 2002).

Programming is of particular concern to educators because it underpins computer sciences, yet is perceived to be a difficult to learn. As a consequence, it is not always an attractive option in schools (Brown *et al.* 2013) and often suffers from progression rates in degree courses (Milne and Rowe 2002; Robins, Rountree, and Rountree 2003).

University computing departments already make more extensive use of technology-enhanced-learning (TEL) tools than institutional norms (Jenkins *et al.* 2011). It therefore appears likely that the aforementioned introduction of new computer science curriculum for primary and secondary education will lead to the development of further learning platforms. Some of these, such as Ceebot (Huber 2008), will be available in modes and variants adapted to specific levels of education. One recent example is the 'Hakitzu' application for mobile and tablet deployment (Clifford 2013). Similarly to Ceebot, Hakitzu exploits the popularity of robotic and gaming themes in education environments.

Why evaluate platforms for learning programming in collaborative settings?

Although immersive environments for learning programming are clearly capable of delivering engaging experiences, it is of key interest to discover how they may be evaluated for their wider educational contribution in terms of collaborative learning opportunities, providing cues for discussion as well as for delivering conventional learning outcomes.

Systems analysts and requirements engineers who specialise in developing platforms that perform in collaborative environments fully understand that technical requirements cannot be isolated from their user-base. Button and Sharrock (1994) maintain that system requirements 'are enmeshed in organisational processes' and 'are not objective; they come from a point of view'. Goguen (1992) similarly states that, 'Requirements are properties that a system should have in order to succeed in the environment in which it will be used'. Tacit and communication processes are therefore widely regarded to be important in guiding the design of systems that must operate in collaborative environments (Heath *et al.* 1995; Jirotka and Wallen 2000).

Similarly, the reliable and stakeholder-representative evaluation of technology-mediated learning is a long-standing concern of many researchers (Hardman and Paucar-Caceres 2010; Jackson 1990; Oliver 2000; Voigt and Swatman 2004). As for the aforementioned requirements engineers, educationists also accept that evaluation of learning platforms cannot take place in isolation of teaching and learning or student experience contexts (Phillips and Gilding 2003). Learning technologies used for teaching computer programming often operate in highly collaborative settings where students are permitted or encouraged to communicate while undertaking practical work. The potential complexity of such interactions extend to social construction and 'appropriation' processes by which users modify behaviours to accommodate technology and even reciprocally adapt technology to uses that were not originally intended by the developers (Overdijk and van Diggelen 2006).

'Mixing' methods to capture collaboration and complexity

In the light of complexities associated with environments for learning programming, a 'mixed-methods' approach is adopted for the investigation reported here. This includes a quantitative treatment of test, self-assessment and questionnaire data to explore student progress, preferences and acceptance of the environment as well as any correlation between these entities. The quantitative evaluation conducted here is combined with deeper ethnographic exploration of the learning space through 'workplace' observation and video analysis.

However, the development of mixed methods as a research paradigm has a turbulent past. Johnson and Onwuegbuzi (2004) consider that this is largely due to the incompatibility of positivist and constructivist philosophies that inform quantitative and qualitative research. The authors highlight how polarities between the two research camps have prevented the development of pragmatic and practitioner-oriented approaches for those '... who would like to see methodologists describe and develop techniques that are closer to what researchers actually use in practice'.

Fully acknowledging that the validity, taxonomy and synonyms of 'mixed methods' are the subject of ongoing discussion (Symonds and Gorard 2008), this study adopts the definition and rationale that mixed methods concern '... collecting, analyzing, and mixing both quantitative and qualitative data in a single study or series of studies. Its central premise is that the use of quantitative and qualitative approaches in combination provides a better understanding of research problems than either approach alone' (Creswell and Plano Clark 2011, p. 5). However, certain writers note that recorded qualitative techniques, such as video analysis, have limitations. Jewitt (2012) cautions that the volume, richness and diversity of information associated with video studies 'can lead to overly descriptive and weak analysis'. Jewitt (2012), after Snell (2011), suggests that 'systematic quantitative analysis' should be coupled with 'micro-ethnographic qualitative analysis', but concedes that there may be practical obstacles to achieving this ideal.

Accepting limitations of video analysis, a 'mixed-methods' viewpoint is generally consistent with the suggestions of Phillips and Gilding (2003) who, citing methodological deficiencies of traditional research paradigms reported by Reeves (1997), state that 'it is more appropriate to try to discover *how* things work in a particular learning context, using a mixture of qualitative and quantitative sources of data'. Phillips and Gilding (2003) further propose that evaluators should adopt 'pragmatic' approaches that are more clearly focussed on questions relating to the effectiveness of ICT on student learning, rather than be driven by a specific methodology that is consistent with a particular research paradigm. Regarding approaches for evaluating the effectiveness of mixed methods, Burrows (2013) reports international consensus among mixed-methods practitioners that standardised guidelines are neither possible nor desirable. The criteria adopted here are therefore guided by a common understanding that evaluation should assess whether there are clear synergistic or integration benefits to method mixing in comparison to independent studies (Burrows 2013).

The overall *aim* of this study is to determine if a mixed-methods approach is effective for exploring learner responses to an interactive TEL environment for computer programming. Such responses include:

- progress with learning and skill acquisition;
- collaborative and tacit communication behaviours;
- the acceptance or appropriation of learning technology; and
- the level to which students engage with the general learning environment.

In doing so, the study also comments on the suitability of technology and the context in which it is delivered.

This aim is addressed using a mixed-methods approach that combines ethnographic techniques, used for assessing collaborative computing environments, with a questionnaire that includes self-assessment and a test of programming understanding.

Methods

The Ceebot platform, used in this study, is specifically designed for learning C-type and object-oriented languages (Huber 2008; Maragos and Grigoriadou 2005). It is based on an animated 3D landscape populated with programmable robotic devices that interact with each other, 'alien' life forms, inanimate objects and representations of human controllers.

Weekly observations were made of a first year BSc Computing module (CS1 Level) to introduce computer programming. Sessions were typically a 1-hour lecture followed by 2 hours of practical work.

The mixed-methods study combined ethnographic approaches, adopted for sensitivity to collaborative and tacit communication behaviours, with a questionnaire that included a self-evaluation of progress and a programming test. The questionnaire provided a means for expanding sample coverage and, as well as corroborating ethnographic findings, also revealed elements that were not addressed in the ethnographic domain.

Two groups of undergraduate students, 39 individuals in total, were observed while working on Ceebot tasks. Ethical procedures were followed to ensure that participants were willing and consented to recordings. The reasons for study, the ownership, protection and the distribution of information were clearly explained. All findings are reported on an anonymous basis.

Ethnographic approach: workplace observations and video analysis to determine how students fulfil tasks at the Ceebot interface

The objective was to explore the actions, events and communications that constituted the collaborative processes by which students approached programming tasks.

Workplace observation

The instructor, in role as investigator, observed problem solving during Ceebot laboratories using a process similar to 'verbal protocols' or 'concurrent thinking aloud' techniques inspired by user-interface research (Kuusela and Paul 2000; Lewis 1982).

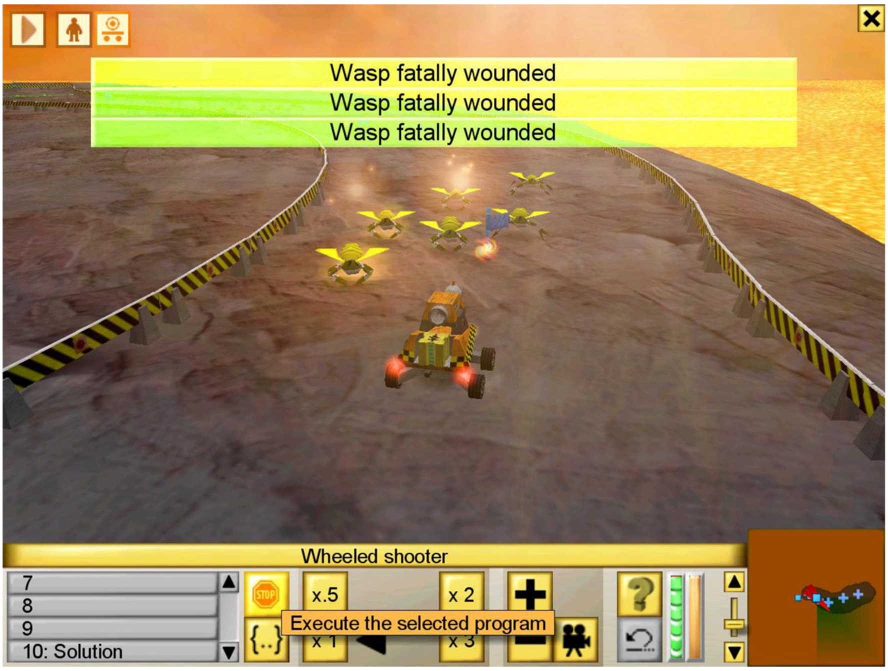
Observations were made during 15 class hours in the final 5 weeks of a 14 weeks module. The overall analytic orientation for this approach is one of ethnomethodology (Heritage 2013; Jirotka and Wallen 2000), insofar that it attempted to understand how students constructed a common understanding of problems and solutions at the Ceebot interface. In the typology of naturalistic research roles adopted by ethnographers (Punch 2014 after Gold 1958), the balance of roles was 'participant-observer' (McMillan and Schumacher 1997).

Video analysis

Video studies extended workplace observations. In a discussion of video ethnography Goldman (2014) expresses that quantitative studies do not 'explain the inside story', thus ethnography is needed to capture 'rich stories that help us understand the meaning of events'. Given that this discourse concerns ethnography for educational research, Goldman's definition of ethnography is very pertinent here; that 'Ethnography is the description, interpretation, and a representation of what researchers experience when on the one hand, ideas and concepts, and, on the other hand, collaboratively constructed artefacts – texts, videos, software – that emerge within a community of practice' (Goldman 2014, p. 25).

Video study was, therefore, used for its sensitivity to 'subtle' and 'tacit' communication processes, and for the convenience of revisiting episodes outside class time. The analytical process followed guidance on the use of video in requirements engineering (Jirotka and Luff 2006).

Video analysis was undertaken for a number of Ceebot tasks during the final 3 weeks of the module. The example presented in Text Box 1 and Figure 1 illustrates the method and the context. Here, students were required to write functions (fragments of code designed to perform discrete tasks) using parameters to program a wheeled robot that pursued and destroyed 'alien insects'. This was part of a suite of tasks intended to introduce software engineering principles of encapsulation and information-hiding (Booch, Jacobson, and Rumbaugh 1999).



Wasp fatally wounded
Wasp fatally wounded
Wasp fatally wounded

Figure 1. Screen capture illustrating the Ceebot environment and successful implementation of code solution shown in Text Box 1 (reproduced with kind permission from Otto Kölbl, developer and distributor of Ceebot).

Text Box 1. Ceebot-analysis task ‘Clear the Race Track’ and example solution.

The race track is infested with aliens. There are packs of Spiders, Wasps and Ants. It needs to be cleared before it can be used. We need a function to destroy a pack of aliens and to make it universal so we can destroy any alien pack … overall, what we need to do is: 1. Destroy all the spiders; 2. Drive on to the next pack (in the direction of the blue flag); 3. Destroy all the wasps; 4. Drive on to the next pack; 5. Destroy all the ants.

This should be possible with just 2 functions: Destroy(…) and DriveOn(…).

Example Solution

```
extern void object::Solution() {
    Destroy(AlienSpider);
    DriveOn(AlienWasp);
    Destroy(AlienWasp);
    DriveOn(AlienAnt);
    Destroy(AlienAnt);
}
//************************
void object::Destroy(int alien) {
    object item;
    aim(0);
    item = radar(alien);
    while(item! = null) {
        turn(direction(item.position));
        fire(0.1);
        wait(0.2);
        item = radar(alien);
    }
}
//************************
void object::DriveOn(int alien) {
    object item;
    while(radar(alien,0,180,0,10) = = null) {
        item = radar(BlueFlag, 0, 180);
        drive(1, direction(item.position)/180);
        wait(0.1);
    }
}
```

A questionnaire survey with integrated self-assessment and test elements

A questionnaire survey was undertaken to corroborate ethnographic findings and to provide wider sample coverage. This comprised a self-assessment of Ceebot progress, a test of understanding and a section concerned with gauging student working behaviour and acceptance of Ceebot. A Likert scale of exhaustive categories was adopted. Categories ranged from 1 for 'strongly agree' to 5 for 'strongly disagree'. Provision for freeform comments was also included with each question, to avoid possible complete foreclosure of response options (Wilson and Sapsford 2006).

Results

Ethnographic approach: workplace observation and video analysis to explore learning behaviour at the Ceebot interface

The classroom environment, a computer laboratory, was behaviourally rich and students were clearly engaged by Ceebot tasks and animation. Visual cues taken from the Ceebot interface were often a catalyst for tacit communication. This provided a focus for discussion and was an 'illustrative' vehicle by which students were more clearly able to articulate problems and solutions to each other and the instructor.

There were notable differences in the extent and type of collaborative relationships. These largely depended on whether students preferred to work individually, in pairs or to network with many classmates. Pairs working most effectively were balanced in ability and contributions to tasks. Few individuals recorded in-code comments, task descriptions and algorithms, despite these being assessed elements. This observation usefully informed teaching practice, prompting the instructor to suggest a model workflow for properly recording tasks and to check that logbooks were regularly updated.

Although of considerable exploratory value, it was difficult for the instructor to function in a dual 'observer-as-participant' role without missing certain subtle and tacit elements of collaboration. For this reason working partnerships between pairs of students were subject to video recording. Two pairs were of particular note for inter-group as well as within-pair communications. One pair, students S1 and S2, were often advised by the other pair, S3 and S4. Students S3 and S4 were notably adept at solving programming problems.

The most striking features of the video study were the authenticity of information, the convenience of revisiting recorded scenarios and the capture of subtle behavioural interactions and tacit communication. One session of 48 minutes duration was notable for the rich complexity of interaction between two students and the lecturer. This demonstrated reciprocal assistance with the use of visual cues from the Ceebot interface. For example, in the 'Dialogue Fragment' (see Text Box 2), S4 conveys his discovery that by reference to a single and general alien-type parameter, only one 'Destroy()' and one 'DriveOn()' function is needed for all types of aliens.

Towards the end, S4 is close to a working solution. Student S3 has continued with his approach of trying to engineer separate functions to recognise each type of alien rather than more desirably referencing aliens as a single and general parameter. S4 overhears S3 querying that the 'general' approach suggested by the instructor will work and joins in to affirm he has successfully implemented a similar solution. This is a three way discussion between S3 and the lecturer (L1 in Text Box 2), with S4 later participating on a tacit, overheard, signal. Both S3 and S4 convey the immediacy and engaging nature of the Ceebot environment in conversation.

Text Box 2. Dialogue fragment (S4's successful completion of the Ceebot task) recorded during in the final 5 minutes of a 48 minute-task.

S4: *{testing code on his 'wheeled shooter' robot}* ... shoot 'em, shoot 'em, shoot 'em ... *{frustrated that the program doesn't correctly execute}* No::ooo!

S3: *{simultaneously talking to L1}* ... but it's not ... not picking up that its finished, it's just saying unknown object, and that should end the while loop ... but it doesn't even see it, it's coming up with an error.

L1: *{trying to steer S3 towards a solution}* The purpose of using ... um ... parameters is that you can pass in alien types ... so what you really want to do is take [S4's] approach ...

S3: ... uh, yeah, I've tried ... tried that first but I don't see any way of doing it ... other than doing three different ... destroy functions *{i.e. one for each alien type}*

S4: *{turning to face S3's screen and RM}* I've, I've done it, I've done it, apart from the fact that the wasps are coming in too early, so they are killing me before I get through to ... (*end*)

The session is followed by S3 stating a preference for Ceebot in comparison with other programming environments, implying that it is a 'safe' platform because it is 'not too complicated' and that 'there is a lot less to go wrong'. This comparison is made with respect to 'Microsoft Visual Studio™', which is a more powerful and complex development platform than Ceebot. Student S3 expands that the IDE (Integrated Development Environment) is intuitive because 'everything is on screen'. Although clearly developing to become a capable and independent programmer, S3 also expresses that he preferred to work collaboratively.

Questionnaires for exploring student perceptions of learning progress and acceptance of the Ceebot environment

Thirty eight students completed questionnaires. Key survey sections (summarised in Table 1 and presented in full in Appendix 1) were:

- the results for self-evaluations of progress against tasks;
- perceptions of subject difficulty;
- a test comprising five questions designed to evaluate understanding of core principles of 'C' programming structures and syntax; and
- a 20-point questionnaire to elicit 'user acceptance', satisfaction and behavioural responses to the learning environment and to Ceebot.

The majority of respondents recorded that they had completed over half of the Ceebot tasks ('Self-Evaluation 1' in Table 1). Most perceived that these were moderately challenging ('Self-Evaluation 2' in Table 1). However, the results of the test in Table 1 indicated that few students fully understood key 'learning outcome' concepts. The popularity and usefulness of collaboration, was clearly

Table 1. Summary of frequencies and distribution for questionnaire and associated self-evaluation and test categories.

SELF-EVALUATION – 1						
(Percentage of Ceebot exercises completed)	<25%	<49	<75	≥75		Total
Frequency	5	8	7	9*		29
SELF-EVALUATION – 2	1	2	3	4		Total
(Perceived difficulty of Ceebot exercises 1: Easy; 2: Moderate; 3: Difficult; 4: V difficult)						
Frequency	2	15*	7	6		30
TEST (number of correct answers to five questions)	0	1	2	3	4	5
Frequency	17*	10	5	4	1	2
QUESTIONNAIRE – CEEBOT PERCEPTIONS	1	2	3	4	5	Total
1: Strongly agree; 2: Agree; 3: Neutral; 4: Disagree; 5: Strongly disagree						
Q1 – It is helpful to discuss with my friends	16*	16*	3	2	1	38
Q2 – It is easy to find Ceebot help information	2	13*	10	11	1	37
Q3 – Animated Ceebot is helpful for understanding	7	17*	8	3	3	38
Q4 – Useful to draft design/algorithms on paper first	8	11*	10	5	4	38
Q5 – Ceebot doesn't help remember concepts	7	4	14*	6	5	36
Q6 – Ceebot is enjoyable	8	13*	11	3	3	38
Q7 – No formal lectures are required	1	9	7	15*	6	38
Q8 – I'd like module to be commercially recognised	8	9*	9*	7	5	38
Q9 – Ceebot graphics are distracting	4	5	9	14*	6	38
Q10 – Unassessed multi-choice questions would help	4	13	15*	2	4	38
Q11 – It is quicker to learn without Ceebot	6	3	14*	9	6	38
Q12[a] – Easiest to complete tasks then cut and paste	10	12*	8	5	3	38
Q13 – Ceebot is good for learning C-type languages	3	14*	11	5	5	38
Q14 – I found helpful websites for completing tasks	1	3	24*	5	5	38
Q15 – Worried that Ceebot may not help get a job	5	6	14*	8	4	37
Q16 – I only work on Ceebot during practical sessions	5	6	9	14*	4	38
Q17 – I need at least 2 more hours to complete tasks	9	20*	3	4	1	37
Q18 – There are too many exercises	7	9	14*	7	1	38
Q19 – I work on Ceebot at home	5	13*	5	7	7	37
Q20 – I'd like to post problems to a forum	5	12*	11	7	3	38

Notes: (a) Q12 abbreviates that it is easier to update logbooks of practical work by cutting-and-pasting code from Ceebot; (b) central tendency (highest frequency of occurrence) is indicated by asterisked values; and (c) the full questionnaire is provided Appendix 1.

indicated by many agreeing that it was helpful to discuss exercises with friends (Table 1, Q1).

There was generally consensus on the following points: (Q2) that it was 'easy' to discover information required to complete exercises; (Q3) that the animated environment assisted with understanding; (Q6) Ceebot was enjoyable; (Q13) that Ceebot is a good platform for learning C-type languages (although most students would have little basis for comparison); and (Q9) that Ceebot graphics were not distracting.

The ambivalent response, 'neither agree nor disagree' that Ceebot 'does not help me remember fundamental programming concepts' (Q5) is consistent with low test marks overall but appears to contradict a general consensus that Ceebot aided overall understanding (Q3). However, although unproven in this study, Q3 may reflect that students believed that they had achieved some level of understanding programming

Table 2. Partial matrix of rank correlations for responses to questionnaire.

	Ex Comp.	Test	Q3	Q5	Q6	Q10	Q11
Exercises completed	1.00						
Test	0.47*	1.00					
Q1	−0.03	0.17					
Q2	0.06	−0.08					
Q3	−0.16	−0.38*	1.00				
Q4	0.14	0.04	0.39*				
Q5	0.41*	0.28	−0.52**	1.00			
Q6	0.02	−0.22	0.49**	−0.30	1.00		
Q7	0.05	0.07	0.10	−0.25	−0.10		
Q8	−0.11	−0.23	0.57**	−0.20	0.42**		
Q9	0.14	0.26	−0.53**	0.37	−0.36		
Q10	−0.20	−0.10	0.49**	−0.47**	0.42**	1.00	
Q11	0.03	0.20	−0.69**	0.47**	−0.57**	−0.33**	1.00
Q12	−0.13	0.06	0.01	0.23	0.12	0.08	0.09
Q13	−0.40*	−0.32	0.64**	−0.66**	0.51**	0.58**	−0.49**
Q14	0.17	0.14	−0.19	−0.14	−0.19	−0.03	0.20
Q15	0.14	0.15	−0.52**	0.23	−0.16	−0.26	0.43**
Q16	0.24	0.41	−0.30	0.49**	−0.10	−0.26	0.28
Q17	0.36	−0.33*	0.22	−0.08	0.23	0.17	−0.19
Q18	0.30	0.16	−0.35*	0.36*	−0.32	−0.41*	0.30
Q19	−0.18	−0.32	0.29	−0.18	0.33	0.11	−0.31
Q20	0.06	−0.01	0.27	−0.24	0.23	0.18	−0.21

Notes: (a) Spearman's r at $p \leq 0.05$ (*) and $p \leq 0.01$ (**); and (b) see Table 1 for explanation of questions Q1–Q20.

while working in the Ceebot environment. A perception of improved comprehension, as implied by Q5, may therefore have developed without necessarily advancing to a level sufficient for confidently articulating subject concepts, such as key programming constructs.

Regarding Ceebot delivery, students favoured formal lectures (Q7), but thought that much time and resources were required to complete tasks (Q16, Q17 and Q19).

Correlation analysis (Table 2) indicated highly significant ($p \leq 0.01$) and positive relationships between responses associated with perceptions of Ceebot's usefulness (Q3 and Q13), enjoyment (Q6) and individual commitment and motivation (Q8 and Q10). The overall direction of significant correlations between positive statements of Ceebot (Q3, Q6 and Q13) with contrasting 'negative' statements (Q5, Q9 and Q11) suggested that questionnaires had been completed with care and diligence.

Although Table 2 correlations with 'test' results were of interest as indicators of learning, interpretation was confounded by a heavily skewed distribution with 70% of the cohort scoring zero or only one mark (see Table 1). Test 'success' was most strongly correlated with completing more exercises, thus associating 'success' with greater practice (Table 2). Support for introducing formative multiple-choice tests (Q10) was significantly correlated with enthusiasm for Ceebot and commitment to the subject (Q3: $r = 0.49$, Q6: $r = 0.42$ and Q13: $r = 0.58$) and disagreement with the converse or opposing statements (Q5: $r = -0.47$, Q11: $r = -0.33$ and Q18: $r = -0.41$).

Perceptions of helpfulness of discussing Ceebot problems with friends (Q1) was found to be correlated with inference that work was also undertaken outside class

(Q16 and Table 2). The absence of significant correlation between Q1 and test score achievement ($r = 0.17$) suggests that partnered work may not have fostered deep learning. However, four of only six individuals who scored three or more correct answers, stated clear preferences to discuss work with friends.

Given the relatively low scores achieved in test results, it was interesting to discover that perceptions of the usefulness of Ceebot's animated learning environment (Q3) were, in fact, significantly correlated with test achievement (Table 2). The negative correlation is simply a function of the direction of the Likert scale (1: strong agreement to 5: strong disagreement).

Discussion

The performance of 'technology' in the context of the learning environment

The investigator and author found that both studies helped to clarify boundaries between areas of human, system and institutional responsibility. Suggestions were noted for extending the specification of Ceebot, such as through providing facilities for tracking learning progress, multiple-choice tests, instructional videos and discussion fora. However, from a system-engineering perspective, Ceebot has a core value and integrity as an immersive, collaborative and engaging focus for education. This integrity is, perhaps, better not diluted with specifications that are more properly the concerns of a wider VLE (Virtual Learning Environment), or for some other pedagogic and organisational intervention.

Two broad categories of learning technology requirement were recognised. These were the 'context' concerns of pedagogic and institutional attention, and other requirements that were identifiable as being purely technical specifications.

Context related requirements inferred and expressed from ethnographic and questionnaire studies included to: (1) maintain a conventional model of a lecture to introduce concepts, followed by laboratory sessions that put theory into practice; (2) reduce the number of exercises; and (3) examine potential for encouraging collaboration as a means for improving the learning experience. A key observation of the video analysis was that the progress of S3 and S4 appeared to be accelerated through communicating their shared discoveries. The value of such 'paired programming' is also widely recognised by Agile Development and Extreme Programming communities (Cockburn and Williams 2000).

Among *technical requirements*, potential revisions included: improving the consistency with which Ceebot automated 'tick' and 'mission accomplished' recording of successfully completed exercises (for both student-formative and lecturer-summative uses); more intuitive help navigation; and configuring Ceebot to run in a reduced 'window' for more convenient copy-paste operations between the IDE and logbook documents. Disregarding aforementioned VLE-type functionality, suggestions more directly associated with Ceebot concerns were the possible inclusion of logbook templates for tasks, and video clips to demonstrate successful task outcomes.

However, the most significant finding of immediate pedagogic concern to the author was the relatively low achievement in the programming test, in spite of an overall belief that Ceebot had aided understanding of programming. The absence of a similarly clear signal that Ceebot also helped with remembering concepts further indicates that the teaching environment requires modification to encourage deeper learning of key concepts.

Having practised and intuitively understood programming with Ceebot, questionnaire returns indicated that few students were able to remember key constructs or to repair code outside the Ceebot environment. As a result there appeared to be general acceptance that formative (unassessed) multiple-choice testing may be a useful vehicle for self-evaluation and for guiding students to areas requiring revision.

The methods and the merits of 'mixing'

The author and investigator found that key advantages of *ethnographic approaches* were insights into communication and tacit behaviours at the Ceebot interface. Verbal protocol methods are sometimes regarded to be vulnerable to reactive influences, where verbalization alters the process under observation, and non-veridicality inaccuracies, such as errors of omission or commission (Russo, Johnson, and Stephens 1989). However, Ceebot observations were believed to be relatively robust because they were recorded concurrently with task activities and because 'thinking aloud' was part of the natural communication in resolving tasks.

The benefits of ethnographic approaches are widely recognised (Heath *et al.* 1995; Jirotka and Wallen 2000) and, for this study, the 'authenticity' of observations outweighed limitations of the ethnographer's dual 'observer-as-participant' role (Punch 2014). The engaging and visual-cue qualities of Ceebot were clearly evident. However, similarly to other researchers reporting surface and deep strategies to process learning materials (e.g. Biggs 1987; Marton and Säljö 1976), it was also apparent that students were perhaps more concerned to complete assessed tasks and logbooks than to spend time inspecting and understanding programming structures. Nevertheless, by entering the learner's domain, the investigator–instructor more fully appreciated that collaboration extended beyond 'information swapping' to shared experiences of problem solving.

Even with options for additional freeform comments, *questionnaire*-based surveys unavoidably constrain participants to responses within a narrow range of expression. However, when interpreted in combination with ethnographic findings, this usefully corroborated the authenticity of certain perceived values, such as of Ceebot and of partnered work. Associated self-evaluations, 'test' assessment and correlations between questions were also directly useful for informing teaching practice and delivery. Findings highlighted qualities associated with test 'success' against learning outcomes, for example greater practice at Ceebot tasks. Results also cautioned that test success was not necessarily associated with collaborative behaviour, Ceebot technology or the immediate learning environment.

The value of collaboration and 'sharing discovery' was clearly evident to the investigator and author from *video analysis* (ethnographic approach '2'), as was associated tacit-communicative behaviour by which students referred to code and took cues from animation at the Ceebot interface. This appeared to reveal a socio-constructivist dimension whereby students had appropriated Ceebot for a collaborative purpose. Such collaborative use was not the explicit intention for the original specification or design of Ceebot. Technological appropriation is a widely recognised phenomenon by which users experiment, evaluate and adopt or reject artefacts according to their resonation with the lifestyles of those users (Carroll *et al.* 2002). Such appropriation is also believed to mediate student behaviours in computer-supported collaborative learning environments (Overdijk and van Diggelen 2006).

Although not immediately beneficial for summative 'test' performance, students nevertheless were able to exercise transferrable discursive and collaborative skills. Such skills are valued in the workplace and are explicitly required of HE-level computing courses (Quality Assurance Agency for Higher Education 2007). Video analysis suggested that it was unlikely that discussion forum technology could adequately substitute the collaborative immediacy, tacit depth, authenticity and richness of Ceebot classroom experiences.

Video analysis also positioned behaviours within the context of pedagogic theory. It was evident that most students responded to the practicality and immediacy of implementing lecture theory in problem solving; this clearly appealing to 'activist' and 'pragmatist' elements of learning style profiles (Honey and Mumford 1982). Recordings also captured aforementioned elements of 'surface', 'strategic' or 'achievement' emphasis on task and logbook completion. This suggested a need for strategies to encourage deeper 'reflective' and 'theoretical' understanding of principles and rule-based approaches to problem solving. Contrastingly, however, there were also recorded examples of students discussing and implementing solutions based on earlier experiences.

Among other significant advantages of video analysis was the ability to revisit events that may have otherwise passed unnoticed. This resonates strongly with Lindsay Jordan's report of findings in 'Research in Learning Technology' in 2012. One participant engaging in video-based reflections on teaching practice notes that when '. . . watching a live presentation, I can switch off or mishear certain points or attach a skewed meaning'. Another, on revisiting a video recording, observed that 'not only were there great suggestions being given that I just hadn't registered at the time (I was too busy writing!); there were also things I'd misinterpreted as I hadn't been able to capture the nuances in what people were saying' (Jordan 2012).

Jordan (2012) also noted that there appeared to be a reduced likelihood of observer bias associated with video analysis, or other confounding losses or distortion of information. These findings are consistent with the merits of video analysis reported by Jirotka and Luff (2006).

In addition to useful corroboration of findings between methods, each method of evaluation proved sensitive to specific facets of the Ceebot learning environment. Overall, ethnographic approaches allowed closer observation of the learner space and collaborative behaviour. Video analysis required greater preparation, but allowed deep inspection of recorded sessions and, thereby, more informed intervention and modification of teaching practice. The questionnaire, with included test, provided broad sample coverage and offered a means to discover subject understanding and student perceptions of their learning environment.

Conclusions

In terms of informing education practice, the most notable findings were that: (1) ethnomethodology, with and without video analysis, confirmed Ceebot's value in providing visual cues, foci for discussion and shared discovery, thereby encouraging productive collaboration and learning behaviours; and (2) the questionnaire and 'test' highlighted that however desirable, collaborative and communicative behaviours alone were not sufficient for deep learning. Analysis of student perceptions indicated that although Ceebot may not have greatly helped in remembering key concepts, it nevertheless aided understanding of programming.

Qualitative ethnographic findings revealed behaviours that suggested high levels of engagement and immersion. Students clearly entered the Ceebot-centred interactive environment to an extent that it was appropriated for a purpose that was not explicit intended, namely a focus for collaboration, communication and shared discovery. By combining ethnographic methods for deep inspection of learner space with quantitative treatment of student test and questionnaire-preference data, it was possible to infer that ethnographic findings (suggesting high levels of acceptance, collaboration, communication and overall engagement) could be extended to the wider cohort.

The qualitative study strengthened confidence that perceptions of desiring to 'work together' expressed in questionnaire returns were based on an authentic appreciation of the value of working collaboratively. However, only the quantitative study of test performance data suggested that other interventions may be necessary to promote deeper learning of programming principles. Nevertheless class observations clearly demonstrated that discussion cues provided by the lecturer were important in directing student dialogues surrounding programming tasks. It is, therefore, possible that by moving lesson emphasis from task completion to a more detailed and reflective discussion of the principles demonstrated by each exercise, the lecturer may thereby encourage greater subject understanding.

Findings, overall, indicate that the mixed-methods use of ethnographic and questionnaire approaches used here acted synergistically to inform teaching practice by revealing the extent to which students benefit from visual and tacit cues provided by interactive learning technology. The degree of authenticity and confidence with which findings could be acted upon was largely attributable to corroboration between the 'mixed' approaches used. Providing methods are carefully selected, applied objectively and combined to address the specific context of the learning technology under study, it appears likely that such approaches have great potential for guiding pedagogical intervention in other TEL circumstances.

References

Barnes, D. J. (2002) 'Teaching introductory Java through LEGO Mindstorms models', *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education*, ACM, New York, pp. 147–151.

Biggs, J. B. (1987) *Student Approaches to Learning and Studying. Research Monograph*, Australian Council for Educational Research Ltd., Radford House, Hawthorn, Australia.

Booch, G., Rumbaugh, J. & Jacobson, I. (1999) *The Unified Modeling Language User Guide*, Addison-Wesley, Reading, PA.

Brown, N. C. C., *et al.*, (2013) 'Bringing computer science back into schools: lessons from the UK', *Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, ACM, New York, pp. 269–274.

Burns, J. (2012) 'School ICT to be replaced by computer science programme', *BBC News*, 11 Jan. Available at: http://www.bbc.co.uk/news/education-16493929

Burrows, T. J. (2013) *A Preliminary Rubric Design to Evaluate Mixed Methods Research*, PhD Thesis, Virginia Polytechnic Institute and State University.

Button, G. & Sharrock, W. (1994) 'Occasioned practices in the work of implementing development methodologies', in *Requirements Engineering: Social and Technical Issues*, eds. M. Jirotka & J. Goguen, Academic Press, London, pp. 217–240.

Carroll, J., *et al.*, (2002) 'Just what do the youth of today want? Technology appropriation by young people', *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, Washington, DC, USA, pp. 1777–1785.

Clifford, G. (2013) 'Hakitzu-promising AI platform blurs lines between gaming and learning', *Wired*, 26 Mar. Available at: http://archive.wired.com/geekmom/2013/03/hakitzu-gaming-learning

Cockburn, A. & Williams, L. (2000) 'The costs and benefits of pair programming', in *Extreme Programming Examined*, eds. G. Succi & M. Marchesi, Addison Wesley, Boston, USA, pp. 223–243.

Cooper, S., Dann, W. & Pausch, R. (2000) 'Alice: a 3-D tool for introductory programming concepts', *Proceedings of the 5th Annual CCSC Northeastern Conference*, Mahwah, NJ, pp. 107–116.

Creswell, J. W. & Plano Clark, V. L. (2011) *Designing and Conducting Mixed Methods Research*, 2nd edn, Sage, London, California, New Delhi.

Department for Education. (2013a) *The English Baccalaureate.* Available at: http://www.education.gov.uk/schools/teachingandlearning/qualifications/englishbac/a0075975/the-english-baccalaureate

Department for Education. (2013b) *Computing Programmes of Study: Key Stages 1 and 2. National Curriculum in England.* Available at: http://www.computingatschool.org.uk/data/uploads/primary_national_curriculum_-_computing.pdf

Department for Education. (2013c) *Computing Programmes of Study: Key Stages 3 and 4. National Curriculum in England.* Available at: http://www.computingatschool.org.uk/data/uploads/secondary_national_curriculum_-_computing.pdf

Fincher, S., Cooper, S. & Maloney, J. (2010) 'Comparing Alice, Greenfoot and Scratch', *SIGCSE'10 Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, ACM, New York, pp. 192–193.

Goguen, J. A. (1992) 'The dry and the wet', *Proceedings of the IFIP TC8/WG8.1 Working Conference on Information System Concepts: Improving the Understanding*, Alexandria, Egypt, pp. 1–17.

Gold, R. L. (1958) 'Roles in sociological field observations', *Social Forces*, vol. 36, pp. 217–223.

Goldman, R. (2014) 'Video representations and the perspectivity framework: epistemology, ethnography, evaluation, and ethics', in *Video Research in the Learning Sciences*, eds. R. Goldman, R. Pea, B. Barron & S. J. Derry, Routledge, Oxon UK, New York, pp. 3–38.

Hardman, J. & Paucar-Caceres, A. (2010) 'A soft systems methodology (SSM) based framework for evaluating managed learning environments', *Systemic Practice and Action Research*, vol. 24, no. 2, pp. 165–185, doi: 10.1007/s11213-010-9182-4.

Heath, C., *et al.*, (1995) 'Unpacking collaboration: the interactional organisation of trading in a city dealing room', *Computer Supported Cooperative Work*, vol. 3, pp. 146–165.

Henriksen, P. & Kölling, M. (2004) 'Greenfoot: combining object visualisation with interaction', *ACM Conference on Object Oriented Programming Systems Languages and Applications*, pp. 73–82, doi: 10.1145/1028664.1028701.

Heritage, J. (2013) *Garfinkel and Ethnomethodology*, Wiley, New Jersey, USA.

Honey, P. & Mumford, A. (1982) *The Manual of Learning Styles*, Peter Honey, London.

Huber, M. (2008) 'Bemerkungen für Lehrpersonen (Notes for teachers)'. Available at: http://www.ceebot.org/index.php?option=com_remository&Itemid=53&func=fileinfo&id=6

Jackson, G. A. (1990) 'Evaluating learning technology: methods strategies and examples in higher education', *Journal of Higher Education*, vol. 61, no. 3, pp. 294–311.

Jenkins, M., *et al.*, (2011) 'The development of technology enhanced learning: findings from a 2008 survey of UK higher education institutions', *Interactive Learning Environments*, vol. 19, no. 5, pp. 447–465, doi: 10.1080/10494820903484429.

Jewitt, C. (2012) *An Introduction to Using Video for Research*, National Centre for Research Methods Working Paper, 03/12, Institute of Education, London.

Jirotka, M. & Luff, P. (2006) 'Supporting requirements with video-based analysis', *IEEE Software*, vol. 23, no. 3, pp. 42–44.

Jirotka, M. & Wallen, L. (2000) 'Analysing the workplace and user requirements: challenges for the development of methods for requirements engineering', in *Workplace Studies: Recovering Work Practice and Informing System Design*, eds. P. Luff, J. Hindmarsh & C. Heath, Cambridge University Press, Cambridge, UK, pp. 242–251.

Johnson, R. B. & Onwuegbuzie, A. J. (2004) ‘Mixed methods research: a research paradigm whose time has come’, *Educational Researcher*, vol. 33, no. 7, pp. 14–26.

Jordan, L. (2012) ‘Bringing video into the mainstream: recommendations for enhancing peer feedback and reflection’, *Research in Learning Technology*, vol. 20, pp. 16–25. doi: 10.3402/rlt.v20i0.19192.

Kölling, M., *et al*., (2003) ‘The BlueJ system and its pedagogy’, *Computer Science Education*, vol. 13, no. 4, pp. 249–268.

Kuusela, H. & Paul, P. (2000) ‘A comparison of concurrent and retrospective verbal protocol analysis’, *American Journal of Psychology*, vol. 113, no. 3, pp. 387–404, doi: 10.2307/1423365.

Lewis, C. (1982) *Using the ''thinking-aloud'' method in cognitive interface design*, IBM Research Report RC 9265, IBM Thomas J. Watson Research Center, Yorktown Heights, New York.

Maragos, K. & Grigoriadou, M. (2005) ‘Towards the design of intelligent educational gaming systems’, *Proceedings of Workshop on Educational Games as Intelligent learning environments, Artificial Intelligence in Education*, University of Amsterdam, Amsterdam, pp. 35–38.

Marton, F. & Säljö, R. (1976) ‘On qualitative differences in learning – 1: outcome and process’, *British Journal of Educational Psychology*, vol. 46, pp. 4–11.

McMillan, J. H. & Schumacher, S. S. (1997) *Research in Education: A Conceptual Introduction*, Longman, New York.

Milne, I. & Rowe, G. (2002) ‘Difficulties in learning and teaching programming – views of students and tutors’, *Education and Information Technologies*, vol. 7, no. 1, pp. 55–66. doi: 10.1023/A:1015362608943.

Oliver, M. (2000) ‘An introduction to the evaluation of learning technology’, *Journal of Educational Technology & Society*, vol. 3, no. 4, pp. 20–30.

Overdijk, M. & van Diggelen, W. (2006) ‘Innovative approaches for learning and knowledge sharing’, *Proceedings EC-TEL 2006 Workshops*, pp. 89–96. Available at: http://dspace.library.uu.nl/handle/1874/25104

Phillips, R. & Gilding, T. (2003) *Approaches to Evaluating the Effect of ICT on Student Learning*, ALT Starter Guide 8. Available at: https://www.alt.ac.uk/sites/default/files/assets_editor_uploads/documents/eln015.pdf

Punch, K. F. (2014) *Introduction to Social Research: Quantitative and Qualitative Approaches*, Sage, London.

Quality Assurance Agency for Higher Education. (2007) *Computing*. Available at: http://www.qaa.ac.uk/Publications/InformationAndGuidance/Documents/computing07.pdf

Reeves, T. C. (1997) ‘Established and emerging evaluation paradigms for instructional design’, in *Instructional Development Paradigms*, eds. C. R. Dills & A. J. Romiszowski, Educational Technology Publications, Englewood Cliffs, NJ, pp. 163–178.

Resnick, M., *et al*., (2009) ‘Scratch: programming for all’, *Communications of the ACM*, vol. 52, no. 11, pp. 60–67.

Robins, A., Rountree, J. & Rountree, N. (2003) ‘Learning and teaching programming: a review and discussion’, *Computer Science Education*, vol. 13, no. 2, pp. 137–172.

Russo, J. E., Johnson, E. J. & Stephens, D. L. (1989) ‘The validity of verbal protocols’, *Memory & Cognition*, vol. 17, no. 6, pp. 759–769.

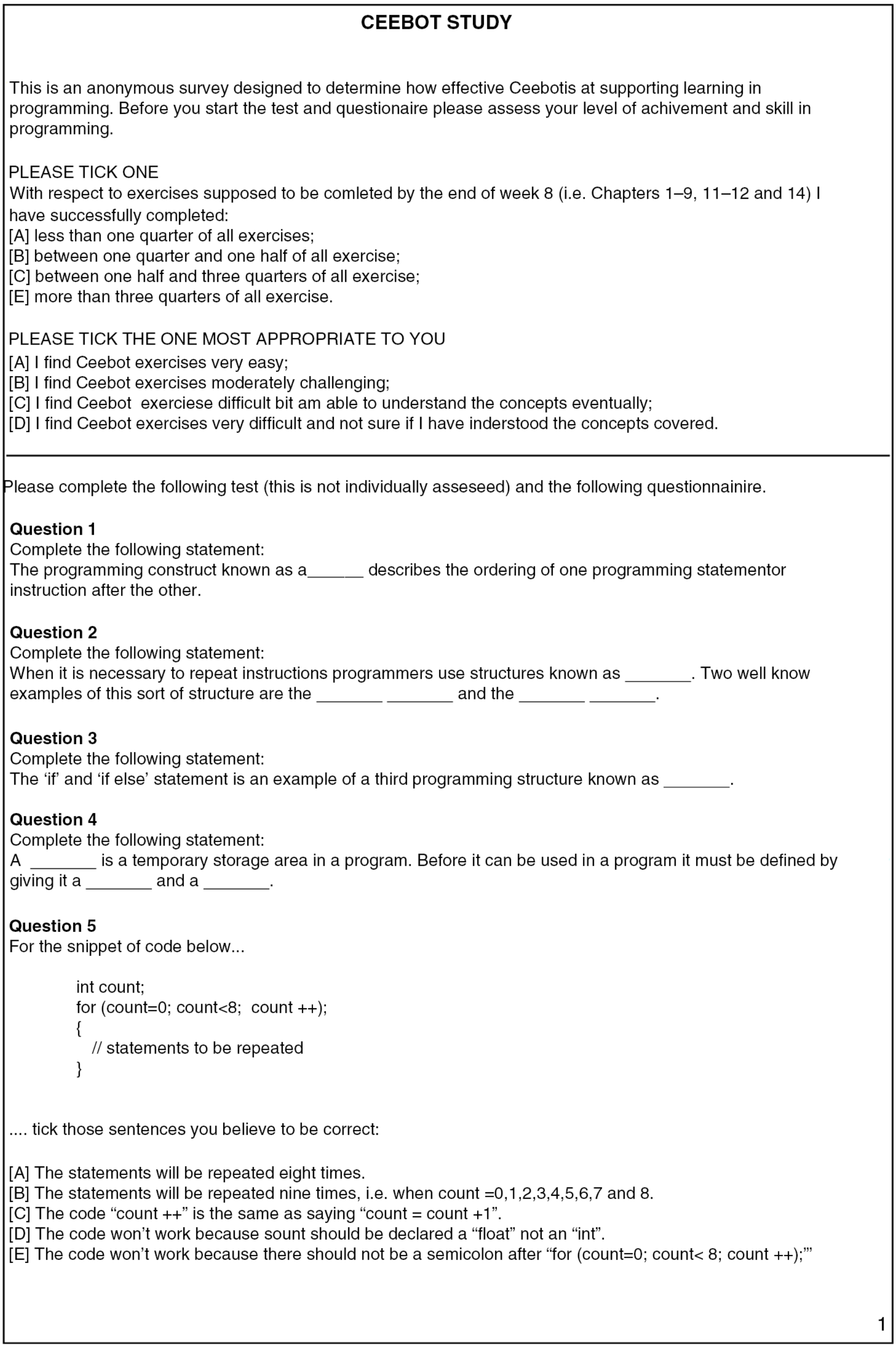
Snell, J. (2011) ‘Interrogating video data: systematic quantitative analysis versus microethnographic analysis’, *International Journal of Social Research Methodology*, vol. 14, no. 3, pp. 253–258.

Symonds, J. E. & Gorard, S. (2008) ‘The death of mixed methods: research labels and their casualties’, paper presented at The British Educational Research Association Annual Conference 2008, Heriot Watt University, Edinburgh. Available at: http://www.leeds.ac.uk/educol/documents/174130.pdf

Voigt, C. & Swatman, P. (2004) ‘Contextual e-learning evaluation: a preliminary framework’, *Journal of Educational Media*, vol. 29, no. 3, pp. 175–187.

Wilson, M. & Sapsford, R. (2006) ‘Asking questions’, in *Data Collection and Analysis*, 2nd edn, eds. R. Sapsford & V. Jupp, Sage, London, California, New Delhi, pp. 93–122.

Appendix 1. Image capture of original questionnaire: front page with self-evaluation and test content.



CEEBOT STUDY

This is an anonymous survey designed to determine how effective Ceebotis at supporting learning in programming. Before you start the test and questionaire please assess your level of achivement and skill in programming.

PLEASE TICK ONE
With respect to exercises supposed to be comleted by the end of week 8 (i.e. Chapters 1–9, 11–12 and 14) I have successfully completed:
[A] less than one quarter of all exercises;
[B] between one quarter and one half of all exercise;
[C] between one half and three quarters of all exercise;
[E] more than three quarters of all exercise.

PLEASE TICK THE ONE MOST APPROPRIATE TO YOU
[A] I find Ceebot exercises very easy;
[B] I find Ceebot exercises moderately challenging;
[C] I find Ceebot exerciese difficult bit am able to understand the concepts eventually;
[D] I find Ceebot exercises very difficult and not sure if I have inderstood the concepts covered.

Please complete the following test (this is not individually asseseed) and the following questionnainire.

Question 1
Complete the following statement:
The programming construct known as a______ describes the ordering of one programming statementor instruction after the other.

Question 2
Complete the following statement:
When it is necessary to repeat instructions programmers use structures known as _______. Two well know examples of this sort of structure are the _______ _______ and the _______ _______.

Question 3
Complete the following statement:
The ‘if’ and ‘if else’ statement is an example of a third programming structure known as _______.

Question 4
Complete the following statement:
A _______ is a temporary storage area in a program. Before it can be used in a program it must be defined by giving it a _______ and a _______.

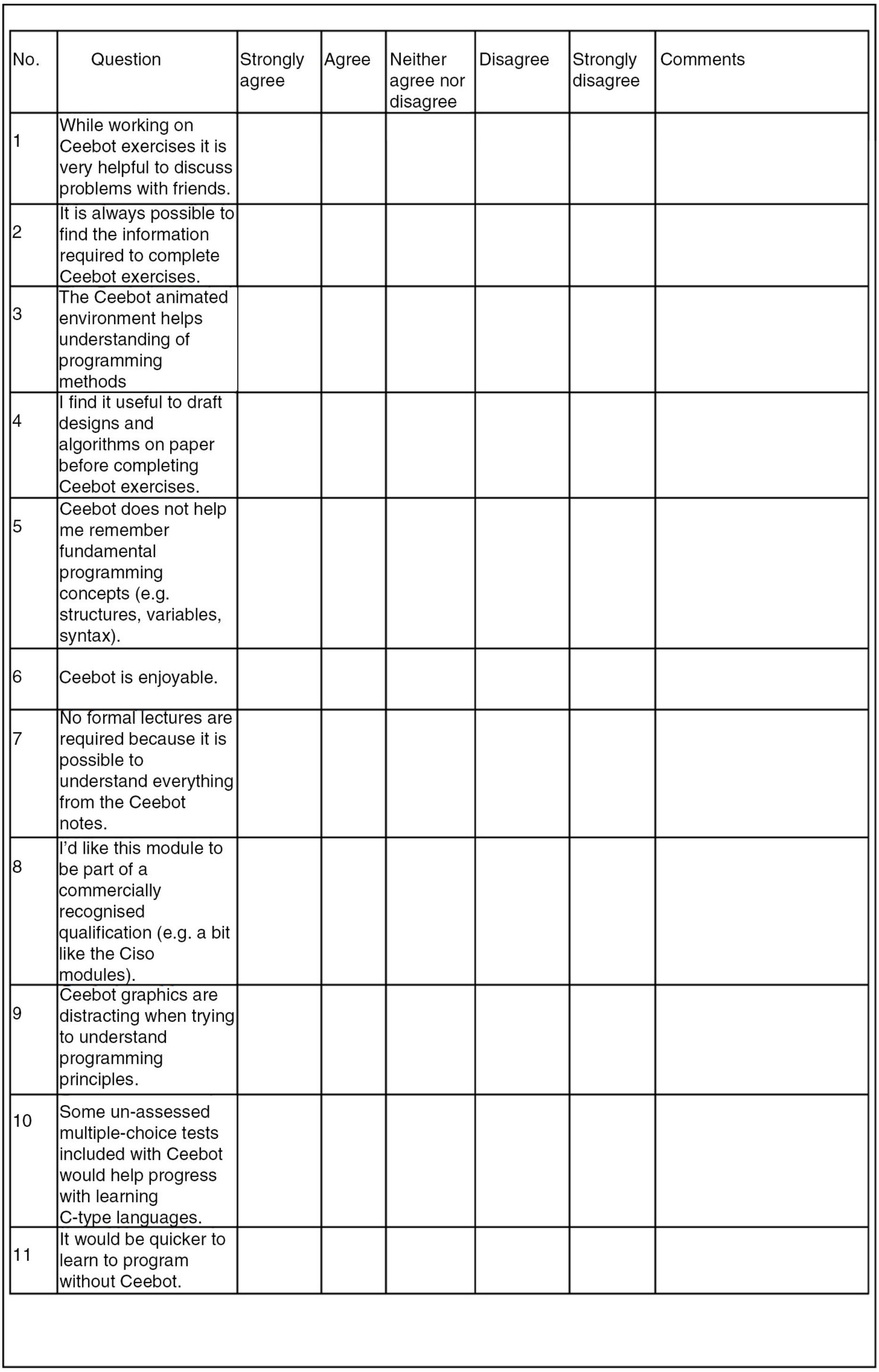
Question 5
For the snippet of code below...

```
int count;
for (count=0; count<8;  count ++);
{
   // statements to be repeated
}
```

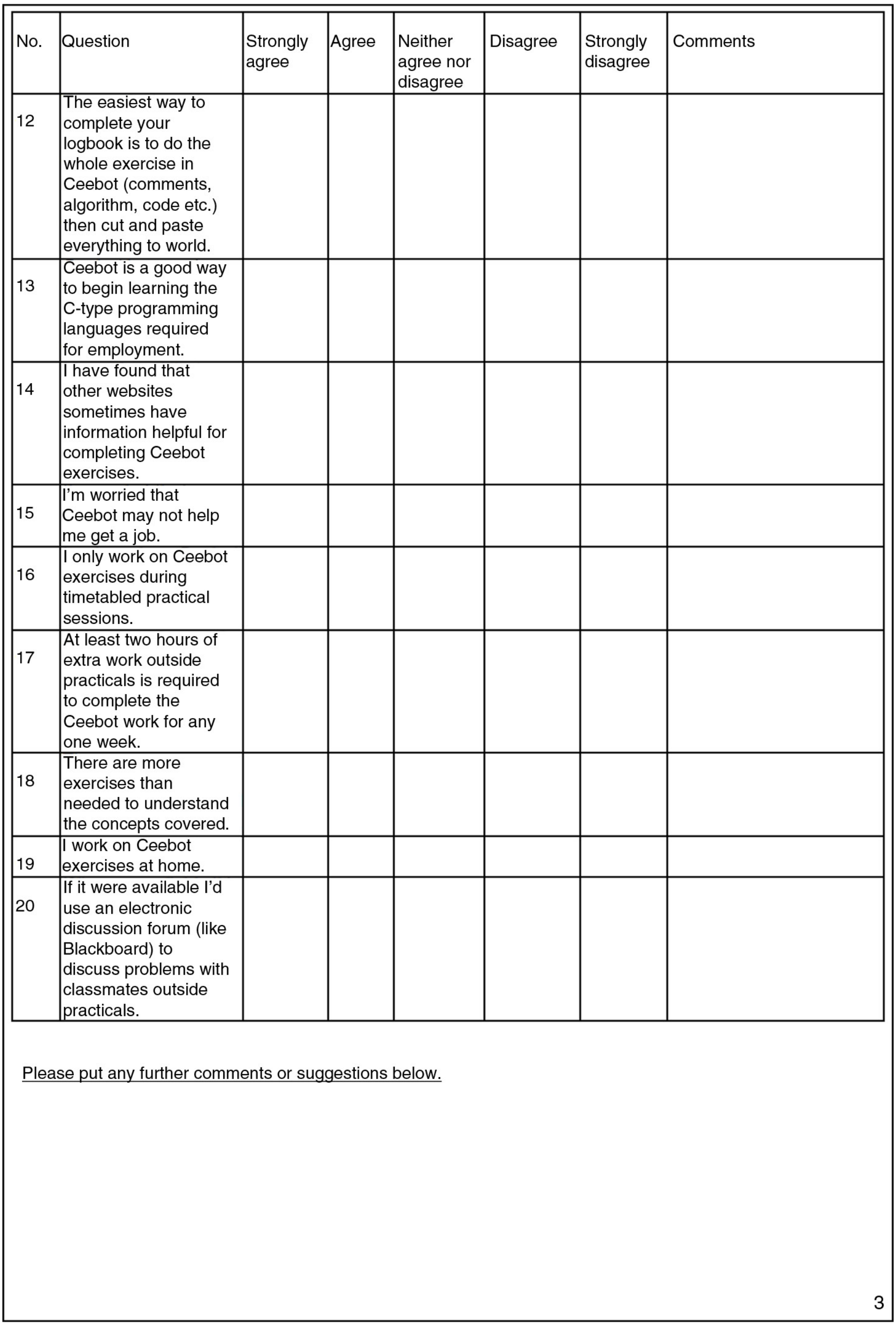
.... tick those sentences you believe to be correct:

[A] The statements will be repeated eight times.
[B] The statements will be repeated nine times, i.e. when count =0,1,2,3,4,5,6,7 and 8.
[C] The code “count ++” is the same as saying “count = count +1”.
[D] The code won’t work because sount should be declared a “float” not an “int”.
[E] The code won’t work because there should not be a semicolon after “for (count=0; count< 8; count ++);”’

1



No.	Question	Strongly agree	Agree	Neither agree nor disagree	Disagree	Strongly disagree	Comments
1	While working on Ceebot exercises it is very helpful to discuss problems with friends.						
2	It is always possible to find the information required to complete Ceebot exercises.						
3	The Ceebot animated environment helps understanding of programming methods						
4	I find it useful to draft designs and algorithms on paper before completing Ceebot exercises.						
5	Ceebot does not help me remember fundamental programming concepts (e.g. structures, variables, syntax).						
6	Ceebot is enjoyable.						
7	No formal lectures are required because it is possible to understand everything from the Ceebot notes.						
8	I'd like this module to be part of a commercially recognised qualification (e.g. a bit like the Ciso modules).						
9	Ceebot graphics are distracting when trying to understand programming principles.						
10	Some un-assessed multiple-choice tests included with Ceebot would help progress with learning C-type languages.						
11	It would be quicker to learn to program without Ceebot.						



No.	Question	Strongly agree	Agree	Neither agree nor disagree	Disagree	Strongly disagree	Comments
12	The easiest way to complete your logbook is to do the whole exercise in Ceebot (comments, algorithm, code etc.) then cut and paste everything to world.						
13	Ceebot is a good way to begin learning the C-type programming languages required for employment.						
14	I have found that other websites sometimes have information helpful for completing Ceebot exercises.						
15	I'm worried that Ceebot may not help me get a job.						
16	I only work on Ceebot exercises during timetabled practical sessions.						
17	At least two hours of extra work outside practicals is required to complete the Ceebot work for any one week.						
18	There are more exercises than needed to understand the concepts covered.						
19	I work on Ceebot exercises at home.						
20	If it were available I'd use an electronic discussion forum (like Blackboard) to discuss problems with classmates outside practicals.						

Please put any further comments or suggestions below.

3