Constructive evaluation: a pedagogy of student-contributed assessment

Andrew Luxton-Reilly* and Paul Denny

Computer Science Department, The University of Auckland, Private Bag 92019, Auckland, New Zealand

(Received 15 October 2009; final version received 31 March 2010)

We present an innovative pedagogical approach that we call constructive evaluation, which shifts students from being consumers of knowledge to participants in a community of peers engaged in actively producing and sharing knowledge. Students are required to author a question that assesses one or more of the learning outcomes of a course. In addition to the question, students write a sample solution. These questions and solutions are stored in a question item bank where they become available for other students to use as a learning resource. Once a student answers a question from the item bank, they can see how other students have answered the question and can reflect on their own response. Additionally, students must review the questions they have answered and are given an opportunity to engage in discussion of questions or answers via a feedback mechanism. In addition to improving content knowledge, students develop important meta-skills such as organising and communicating knowledge; judging the quality of information; giving and receiving feedback and improving self-assessment skills. This approach is aligned with both reflective professional practice and social theories of learning.

Keywords: constructive evaluation; peer assessment; contributing student pedagogy; question-posing

1. Introduction

The development of content knowledge in computer science is essential. However, in the knowledge era characterised by unprecedented access to information and rapidly changing technology, students need a host of other skills to operate effectively (Assister, 1995; Stephenson & Yorke, 1998). Two of these skills, the ability to make critical judgements about the quality of information and the ability to communicate effectively, are both fundamentally important.

*Corresponding author. Email: andrew@cs.auckland.ac.nz
Collis and Moonen (2005) note that the information age requires students to develop a new set of skills, distinct from that of the industrial age. Individuals have to develop social skills in order to work effectively as members of multi-faceted teams, interact regularly in professional communities and adapt to changing environments. The ACM Computing Curricula Overview Report (Shackelford et al., 2006) reinforced the importance of meta-skills for Computer Science graduates by stating that any reputable computing degree program should include:

The identification and acquisition of skill sets that go beyond technical skills. Such skill sets include interpersonal communication skills, team skills, and management skills as appropriate to the discipline. To have value, learning experiences must build such skills (not just convey that they are important) and teach skills that are transferable to new situations. (p. 36)

The software industry relies on the reuse of code to maintain quality and productivity. Consequently, developers must identify and evaluate reusable assets to select the most relevant to the task at hand (Mili, Mili, & Mili, 1995). Before assets can be identified and evaluated, programmers need to reflect on their requirements and formulate questions that become the basis for evaluation. The ability to construct appropriate questions, and determine how to best assess the quality of a solution is a core skill for a programmer working in industry. As educators, we need to encourage students to develop these skills.

In this article, we present an innovative pedagogical approach that we call constructive evaluation, which shifts students from being consumers of knowledge to participants in a community of peers engaged in actively producing and sharing knowledge. Students are required to create assessment questions, review and evaluate questions written by other students, write solutions and review and evaluate the solutions of others. In addition to improving content knowledge, students develop important meta-skills such as organising and communicating knowledge; judging the quality of information; giving and receiving feedback; and improving self-assessment skills. This approach is aligned with both reflective professional practice and social theories of learning.

2. Theoretical framework

Constructivism is a theory of learning which claims that people do not passively receive knowledge transmitted from a teacher, but instead they construct knowledge and meaning from their own experiences of the world (Ben-Ari, 2001). Constructivism is now considered to be the dominant theory of learning in science education (Matthews, 1993).
The approach described in this article is framed by *social learning theory* and *social constructivism*, a form of constructivism that emphasises social interaction in the construction of knowledge. Constructive evaluation shares with both *communal constructivism* and *contributing student pedagogies* an emphasis on the creation of resources for the learning community. By focusing on judgements about the quality of information, we encourage the development of skills that underpin life-long learning.

### 2.1. Social learning theory

The social learning theory developed by Bandura (1986) claims that people learn by observing others, with the environment, behaviour and cognition all as the chief factors in influencing development. Students learn by modelling the behaviour of others, and also by seeing the behaviour of others being reinforced or punished.

Constructivist theory emphasises that knowledge is constructed by integrating prior experience with new information obtained through *involvement* in activities. However, social cognitive theory suggests that *vicarious experience* is also a valuable source of new information. Bandura’s work illustrates how the observation of others is itself an experience that can contribute to the construction of knowledge. Observing how other students solve problems and the feedback (both positive and negative) that they receive on their solutions provides valuable learning opportunities.

### 2.2. Social constructivism

The theory of social constructivism advanced by Vygotsky (1978) argued that knowledge is constructed both internally, and externally through social interplay. Interaction with others is critical to the process of learning and the construction of meaning. Since all students have different prior experiences, they are likely to construct different meanings when exposed to the same information. Through discussion and social interaction, students arrive at shared meaning and can develop deeper understanding.

Vygotsky believed that learning occurs when a child is working within their zone of proximal development. The zone of proximal development describes tasks that the child cannot complete on their own, but could complete with the help of someone who is more competent at performing that task. Vygotsky’s social constructivist theories support cooperative learning pedagogies in which peers help each other learn.

Although the traditional medium for social interaction is face-to-face, social learning also occurs online in environments in which learners express a willingness to share information with others and join others in virtual learning experiences (Alexander & Boud, 2001; Mayes & Fowler, 2006).
### 2.3. Communal constructivism

The term *communal constructivism*, coined by Holmes, Tangney, FitzGibbon, Savage, and Mehan (2001), describes an approach derived from social constructivism in which knowledge is actively constructed by peers for the purpose of sharing with their learning community. Holmes et al. (2001) argue that information technology has expanded our capacity for communication, and the ability to share and archive information effectively. The synergy between social constructivism and online forms of communication can be leveraged to share socially constructed meaning beyond the immediate actors involved in the construction of knowledge to the wider learning community. They define communal constructivism as:

... an approach to learning in which students not only construct their own knowledge (constructivism) as a result of interacting with their environment (social constructivism), but are also actively engaged in the process of constructing knowledge for their learning community. (p. 3114)

Leask and Younie (2001) explore the idea of communal constructivism further. They claim that communal constructivism can be distinguished from other social constructivist approaches in three fundamental ways: firstly, communal building of knowledge is emphasised; secondly, knowledge is constructed by connecting with real communities that have specialist knowledge throughout the world, and thirdly, technology allows knowledge to be obtained, adapted and shared by a wide community of learners.

Our own approach shares with communal constructivism an emphasis on the communal construction of knowledge for the learning community. However, communal constructivism views the learning community as an international one constituted by subject experts and students alike. We take a more localised view and consider the learning community of our students to be contextualised to a single course.

### 2.4. Contributing student pedagogies

Hamer et al. (2008) define the concept of *contributing student pedagogies* and illustrate how they apply to Computer Science. A Contributing Student Pedagogy is defined as:

A pedagogy that encourages students to contribute to the learning of others and to value the contributions of others. (p. 195)

This broad definition encompasses a variety of different approaches that typically have the following characteristics: shifting students from passive to more active roles with greater responsibilities for learning; a focus on
content contributions; assessing the quality of the contributed materials; encouraging the development of learning communities and the use of technology to facilitate student activity.

Constructive evaluation can be categorised as a contributing student pedagogy in which students contribute questions and answers modelled on traditional assessment items (examination-style questions) as a shared learning resource.

2.5. Learning for life

In the information age, where changes in technology are rapid and have a significant impact on the activities of Computer Science graduates, learning content knowledge alone is insufficient. Successful graduates need to develop skills that allow them to adapt and learn throughout their careers (Assister, 1995; Shackelford et al., 2006; Stephenson & Yorke, 1998). The skills that underpin lifelong learning, such as identifying appropriate criteria and making judgements of quality, are skills involving assessment (Boud, 2000).

The main goal of higher education has shifted from making students knowledgeable within a specific domain to transforming students into ‘reflective practitioners’ who can reflect on their own practice and engage in lifelong learning (Dochy, Segers, & Sluijsmans, 1999; Schön, 1983). As educators, we need to develop students’ ability to assess their own learning and generate their own feedback, both of which encourage self-regulated learning (Nicol & Macfarlane-Dick, 2006). Boud (1995) goes further to suggest that the process of self-assessment is critical for learning, and courses that do not encourage self-assessment can actually undermine lifelong learning.

Brookfield (1985) notes that the advice, skill modelling and information provided by other students are also crucial elements of independent lifelong learning. He states that “The learning activities of successful self-directed learners are placed within a social context, and other people are cited as the most important learning resource” (p. 9).

Studies of self-assessment (Boud & Falchikov, 1989; Dochy et al., 1999; Falchikov & Boud, 1989) and peer assessment (Topping, 1998; Boud, Cohen, & Sampson, 2001) have shown significant benefits for students, including increased student confidence in their own ability, growth in competence, increased responsibility for one’s own learning, more accurate judgements of quality (both of one’s own work and that of others), improved problem-solving skills and increased student satisfaction.

To develop lifelong learning skills essential for success in modern society, students need to be engaged in social learning activities focused on assessment. The pedagogy of constructive evaluation described in this article contributes to this goal.
3. Constructive evaluation

Constructive evaluation is a pedagogy focused on the construction and evaluation of assessment items. Students are required to author a question that assesses one or more of the learning outcomes of a course. In addition to the question, students write a sample solution. These questions and solutions are stored in a question item bank where they become available for other students to use as a learning resource.

Authoring the questions is only part of the process. Students are also required to answer questions from the item bank (i.e. questions submitted by other students). Once a student has answered a question, the answers submitted by other students are revealed. The student can see how other students answered the question and reflect on their own response. Additionally, students must review the questions they have answered and evaluate the quality of questions and answers. They also have the opportunity to engage in discussion of questions or answers via a feedback mechanism.

Each aspect of the process is described in more detail below.

3.1. Authoring questions and sample solutions

The process of authoring an assessment question provides numerous learning opportunities. Students begin by using the learning outcomes of a course to design their questions. Learning outcomes are typically expressed as general descriptions of performance criteria while assessment questions are expected to be framed in the context of the teaching material used in a given course. By developing questions that assess course material, students are engaged in a learning activity that raises awareness of what is important in the course.

For example, a learning outcome for a standard first-year programming course taught in Java might be “Learners will be able to evaluate boolean expressions”. A student writing a question about this topic would refer to the course material to find examples of appropriate boolean expressions and determine which boolean operators were discussed in the course. They might reflect on the course notes and consider the order of precedence of the boolean operators and possible confusion between the && and || operators. The question they develop should assess the specifics of the course material according to the criteria specified by the learning outcome. In this context, a suitable question might be:

\[ \text{If } a = \text{true and } b = \text{false, what does the expression } b \& \& b || a \text{ evaluate to?} \]

While students are writing questions, they learn to express themselves in writing using clear, unambiguous language. The skill of asking clearly
specified questions is important to practice, since it will help students to learn more effectively when they apply it to topics they do not fully understand. Asking for help is more likely to result in a positive outcome when the question is clear and unambiguous.

The main benefits of authoring questions are to engage students in activities that focus on the content of a course, and to encourage them to develop self-assessment skills so they can evaluate their own understanding of course content with respect to the defined learning outcomes.

Before students can write a question and solution, they need to understand the relevant content. In some cases, the process of trying to author a sample solution may expose a lack of understanding about a topic.

Authoring a sample solution requires students to express their answer with clarity. Other students will be using the solution, and will be judging the quality of the solution, so it needs to be explained in a way that is accessible to a novice audience. The main benefits of authoring sample solutions are to understand content knowledge and communicate effectively about content knowledge to others.

Students would typically submit their questions electronically.

3.1.1. Multiple-choice questions (MCQs)

In the case of a multiple-choice question, students would have to generate an appropriate question stem, and provide a number of suitable alternatives. The act of generating the alternatives is perhaps more interesting than the question stem, since the best alternatives are ones that reveal a possible misconception.

Following the example of a boolean expression given earlier, a student might identify that there are two possible sources of error: incorrect order of precedence, and mistaking && for the logical or operator and || for the logical and operator. In that case, they could develop a question that evaluates the learning outcome as follows:

What do the two expressions false && false || true, and true || false evaluate to?
(a) true and true
(b) false and true
(c) false and false
(d) true and false

Students who evaluate the first expression as being false have misunderstood the meaning of the operators, or they have misunderstood the order of precedence. Students who evaluate the second answer as false have clearly misunderstood the meaning of the || operator. A student who
created this question would have had to think about the possible mistakes that a learner might make. Explicitly documenting the pitfalls and misconceptions as well as the correct answers engages students in a deep and meaningful way with the subject matter.

Since the question includes the answers, students would normally be expected to explain why the designated alternative is the correct one. If the alternatives are well-chosen to expose misconceptions, the author of the question may choose to describe the misconceptions and explain why they lead to incorrect answers.

This process encourages students to think deeply about the topic, and consider many different variations, along with appropriate justification for those variations.

3.1.2. Free-response questions

In this article, the term free-response is used to describe a written answer, generated by a student, that is one or more words in length. Questions that require free-response answers are perhaps easier to construct than MCQs since there is no need to explicitly state the possible misunderstandings. However, given that the answers will be of indefinite length, it is important to frame the question so that it can be answered in a reasonable time frame.

The sample solution authored by a student provides an opportunity to express concepts in written language, a skill that is often overlooked in Computer Science. The ability to communicate effectively about technical details is critically important for a computing career.

3.1.3. Programming questions

Programming questions are a special case of free-response question with particular relevance to the Computer Science Education community.

Programming questions fall into a small number of patterns. In traditional, instructor defined questions, students are typically asked to: trace some existing code and state the output; figure out the purpose of some code (either to explain in plain English or to produce the output of a program that is too complex to trace, so they are required to understand the purpose before they can produce the output); or, they are asked to write some code that fulfills a given set of specifications.

If a student writes a question that requires a written program as a solution, then they should write clear specifications for the program, normally including an example of input and output (i.e. a test case). A question that omitted such details would be ambiguous and difficult to answer. If a student writes a question that requires tracing to produce an output, they should write correct code that performs a given task. Writing
a correct program of the appropriate length and complexity requires students to think carefully about their audience and consider the clarity of their program structure. All of these activities engage students in independent learning, and require students to think about how to communicate to others about code.

### 3.2. Answering questions

One of the major advantages of this pedagogy is that a large item bank of questions is developed by students with very low cost to teaching staff. This item bank of assessment questions can be used by students to practice their skills and revise course material. Writing answers to questions gives students an opportunity to apply the knowledge and skills they have gained in a course. However, instead of the questions being obtained from an authoritative source, the questions are generated from the student cohort.

When the questions are student-generated, they are frequently less precise than instructor-generated questions. This ambiguity is perhaps not ideal for use in a time-limited formal assessment such as an examination, but may even help students to clarify their own thinking when it occurs in a learning context. Students must think deeply about the meaning of the question with respect to the course material. In cases where the question is ambiguous, students must weigh up the different interpretations and make a decision as to how to interpret the question. Making these decisions encourages students to think deeply about the topic at hand.

The audience for the answer is not an instructor, but like all the other material produced throughout this pedagogy, is the community of student peers. Other students will review and rate the solution produced in response to a given question. Since other students are rating the solution, it is important that the solution is expressed as clearly as possible.

### 3.3. Rating questions and answers

The process of evaluating and rating work produced by their peers is an important component of this pedagogy. Students are required to evaluate the work of their peers, rating both questions and answers that they view. The act of evaluating work serves dual purposes. It provides essential feedback to students operating in a learning environment in which there is no authority to define quality, and it encourages students to engage in activity at a high cognitive level.

The ratings provide feedback to the author about the quality of their work. They can also be used by the community to determine which questions have high quality and are worth responding to, and which answers are of high quality and should be attended to. Observing the differences between high rated answers and low rated answers can be a
valuable experience to learn what makes an answer a good one. Observing the ratings given by others can validate a students’ own rating. The act of providing a rating may encourage students to think carefully about why a question or answer is good. What makes a piece of work high quality? What standards should they apply? The development of standards and criteria on which to make judgements of quality is a critical component of life-long learning. In the information age, where the Internet is a central source of information for technology, and peers play a critical role in information dissemination, the ability to judge the quality of information is essential.

3.4. Discussions and formative feedback

Once a student has reflected on their answer to a question, and has evaluated the question’s relevance and quality, an opportunity for further discussion needs to be provided. The ability to write open-ended feedback, or comments, can serve a number of complimentary purposes.

If a student is left puzzled by a question, particularly one that appears to have been successfully answered by most other students, they may want to reach out to their peers and ask for clarification. Students reading such a request can be adept at providing help as they themselves may have recently had a similar misunderstanding and therefore can have a unique perspective for how that can be resolved.

Having students provide open-ended feedback on one another’s questions can also help to improve the overall quality of the repository. While the ratings assigned to questions provide a coarse-grained measure of quality that can help direct student study efforts, the comments associated with a particular question can directly point out errors in the construction of that question and can include suggestions for improvement. This feedback can be used directly by the question author to improve their question. When other students read comments that provide valid, constructive criticism, they can glean useful tips and ideas for developing their own questions. If a question is accompanied by an explanation that is poor quality or insufficient, students who feel they can improve on the explanation can write their own. In doing so, not only does the usefulness of the question increase, but a student writing an improved explanation may reinforce their own understanding.

Positive reinforcement also occurs by allowing students to comment on their peers’ questions. If students find a particular question very useful, they are able to praise it, which in turn may encourage the author to contribute more high quality questions. In general, discussion and feedback support the social interplay and communication which are vital components of the social constructivist theory on which this pedagogy is based.
3.5. Related work

Although many instructors may have had some experience asking students to prepare sample questions in a traditional pen-and-paper environment, most such activities fall short of providing students with the rich learning experience provided by constructive evaluation. There are numerous teaching strategies that involve students in the development of questions (e.g. Balajthy, 1984; Fellenz, 2004). In some reported cases, the completed questions were shared with other students, but not peer reviewed (Arthur, 2006; Horgen, 2007). In other cases, questions were authored, shared and evaluated by students in an approach akin to constructive evaluation (Barak & Rafaeli, 2004; Chang, Huang, Tung, & Chan, 2005; Yu, Liu, & Chan, 2002). However, none of these pedagogies is identical to our own. They require significant instructor involvement, have not been applied in large classes, and have not been used in the context of computer science.

4. Implementation

Our previous analyses of constructive evaluation have been in the context of first-year programming classes in Computer Science and Software Engineering, and we summarise those findings in section 5. These courses teach introductory Java and introductory MATLAB and C programming, respectively. We felt that constructive evaluation was an appropriate pedagogy for these classes, as it provided students with regular opportunities for self-testing and reflection. This addresses the call made by Murphy and Tenenberg (2005) who empirically investigated the extent of Computer Science students’ knowledge of their own learning, and concluded that, “students should be provided with regular opportunities for empirical validation of their knowledge as well as being taught the metacognitive skills of regular self-testing” (p. 148). At this time, we have personally taught more than 10 courses that have used constructive evaluation in Computer Science.

Students are typically introduced to idea of constructive evaluation by presenting the reasoning behind the approach and the potential benefits to those students that actively engage with the pedagogy. A sample of material produced by students is used to illustrate good question design, and the technology used to support the pedagogy is demonstrated. We have noted that student interest is sustained at higher levels if a small sample of the student-generated questions are presented during traditional lecture times. This not only encourages participation but also acts to showcase good questions. The instructor is provided with an opportunity here to discuss the alignment of questions with learning outcomes. We would recommend that once a week, the highest rated
student question is presented to the class and the author of the question is invited to receive a small reward (e.g. a “chocolate fish” or small candy bar).

We have found that most students do not voluntarily write questions. We therefore suggest that a small portion of the grade for a course be assigned to writing questions (e.g. 1% of the final grade is obtained by writing two questions). We have noticed no real difference in the quality or amount of contribution that students make when the percentage of grade is varied between 1% and 7% for writing questions. However, no assigned grade is necessary for students to answer questions, since most students use the question repository for revision purposes on a voluntary basis.

Although we have no personal experience, instructors who have used our purpose-built tools for constructive evaluation and imposed a high contribution requirement (e.g. two new questions per week) report negative feedback from students who felt that the high workload requirement was unduly burdensome. We suggest instructors who wish to introduce a high level of question contribution from students consider a compensatory decrease in existing course workload.

4.1. Providing appropriate guidance

Since students are usually unfamiliar with the task of constructing assessment items, they should be supported with sufficient guidance to enable them to perform the task adequately. One way to provide this scaffolding is to discuss the construction of assessment items in class (Fellenz, 2004; Palmer & Devitt, 2006). This may also serve the purpose of focusing student attention on the learning outcomes of a course, discussing in detail the aims and goals of the course and how they might be assessed.

Exemplars of assessment items should also be provided to students, allowing them to identify and model appropriate tasks. Typically, students have access to many such exemplars through exercises and tests conducted within the classroom, in laboratory settings, in textbooks, or archived from examinations used in previous years.

Students should be actively encouraged to provide feedback to their peers who are developing assessment questions. Since their peers are working on the same tasks and understand the difficulties learning particular topics, they may be more effective than an instructor at helping a student work in their zone of proximal development. Certainly the amount of feedback and opportunities for modelling behaviour provided by their peers is far greater than would be possible for an instructor to provide. Instructors can explicitly encourage such feedback by showcasing examples of desirable feedback in lectures, or by providing a scoring mechanism that rewards valued feedback.
4.2. The use of technology

Constructive evaluation requires the questions and answers created by students to be evaluated by others in the class. The questions, answers, evaluations and discussions are all visible to the entire learning community. For vicarious learning to occur, it is essential that the evaluations and discussions occur publicly. If this pedagogy were to be implemented using traditional teaching media, very careful planning would be required to ensure that evaluations, aggregated responses and formative feedback were managed in a timely fashion.

While the use of technology is not necessary, the management and administration required to collect and distribute questions, answers, formative feedback and ratings on a regular basis would be prohibitive in a large class. In small classes, a limited implementation of constructive evaluation may be possible. For example, Fellenz (2004) reports on a small course in which students peer-reviewed MCQs, and although the activity was not supported by technology, students claimed that the development of the MCQs helped develop a deep understanding of material.

The choice of technology is important in order to fully exploit the benefits of the pedagogy. Generic software, such as learning management systems (LMSs), can be used with some success as reported by Horgen (2007). However, our early experience with generic software (an online forum) mirrors that of Horgen. We found that discussions were well supported, but the ability to effectively evaluate questions and aggregate responses was lacking. We believe that constructive evaluation is best supported through purpose-built tools.

5. Results

We developed a purpose-built tool, PeerWise, to support the pedagogy of constructive evaluation, and have previously reported how it is used by students. We discovered that students believe both writing questions and answering questions helps them to learn (Denny, Luxton-Reilly, & Hamer 2008a). Although they believe that writing questions helps them to learn, most students do not write questions unless they are compelled to (e.g. if there are marks allocated for question writing). However, students make extensive use of the system for answering questions in drill-and-practice fashion during the study period prior to the final exam (i.e. after students have completed the compulsory tasks associated with constructive evaluation – the authoring and answering of questions – they voluntarily use the material contributed by other students for revision before the final exam). This provides a clear indication that students value the material as a good study resource. They enjoy seeing
student-contributed material, and particularly like reading and writing feedback. In short, students valued the communal learning experience.

Perhaps most compellingly, we found that students who used PeerWise more actively throughout a course had higher exam scores (both in the multiple choice section and in written sections) than those who used PeerWise less actively. This result suggests that active use of PeerWise may result in deeper learning (Denny, Hamer, Luxton-Reilly, & Purchase 2008b).

Another recent study (Denny et al., 2010) showed that students who were most active answering questions on PeerWise improved their rank in class relative to those students who were least active, and that this effect held over students of different ability levels. In other words, students who are engaged with the pedagogy of constructive evaluation in these classes perform better than those who are not.

The majority of students using the system in a first-year programming course wrote clear, error-free questions with feasible distractors and a correct answer. The vast majority (95%) of the questions involved code in either the question stem or the distractors, consistent with the typical format of instructor created questions appearing on the course’s final exams. The weakest aspect of the student-generated questions were the explanations, 43% of which were rated by instructors as being of poor quality. However, feedback provided through the peer review process meant that even poor questions could provide good learning opportunities (Denny, Luxton-Reilly, & Simon 2009a). A study of a typical first-year programming course showed that the student-generated question data bank covered all the major topics in the curriculum (Denny, Luxton-Reilly, Hamer, & Purchase 2009b). It included complex questions involving more than one topic, and questions spanning a range of difficulty were observed for all topic areas. The question data bank was of an acceptable level of quality (Purchase, Hamer, Denny, & Luxton-Reilly, in press).

One frequent concern of instructors is that student generated content may contain errors and may, in some cases, mislead students. It is worth noting that almost all teaching material occasionally contains errors (including textbooks), however, students are well aware that the questions and answers they are viewing are contributed by their peers, and the pedagogy of constructive evaluation explicitly encourages students to evaluate the material they encounter. We checked a sample of the questions in a typical student-generated repository and found 11% contained errors. In every case, a student had detected the error and provided feedback in the discussion thread (Denny et al., 2009a).

We want students to be critical about the quality of material they encounter. The pedagogy of constructive evaluation is designed in part to
prepare students to deal with the massive amounts of information encountered in the information age. Although we need to continue to monitor the way that students engage with these activities, our findings to date suggest that instructors need not be overly concerned about the impact of errors in student-generated content.

Two examples are shown below, to illustrate student generated questions and examples of feedback comments. These questions were selected from a repository of 2627 questions (approximately 4.4 questions per student) developed by a class of 601 first-year engineering students learning introductory MATLAB and C programming. The repository was examined after the completion of a 12-week course. At the time of the final exam, students had submitted 77,257 answers (approximately 128 answers per student). Given the size of the repository, it is difficult to select example questions that are representative of the questions typically found in the data bank, but nevertheless the two questions shown below have some points worthy of discussion.

5.1. Example 01
5.1.1. Question

```c
#include <stdio.h>

int main(void)
{
    int a, b, c, d;
    a = 4;
    b = 5;
    c = 2;

    a = b / c;
    b% = 2;
    c--;           
    d = a + b + c;
    printf( "%d", d);

    return 0;
}
```
What is displayed on the screen?

5.1.2. Alternatives

The alternatives created by the author are shown in Table 1, together with the percentage of students who selected each alternative.
5.1.3. Explanation

As all variables are of type int

\[
\begin{align*}
    a &= 4 \\
    b &= 5 \\
    c &= 2 \\
    a &= b \div c = 5 \div 2 = 2 \text{ with } 1 \text{ remainder } = 2 \\
    b \% &= 2 = 5\div2 \text{ is } 2 \text{ with } 1 \text{ remainder } = 1 \\
    c &= 2-1 = 1 \\
    d &= a + b + c = 2 + 1 + 1 = 4
\end{align*}
\]

5.1.4. Formative feedback and discussion

The comments created by students who answered the question are reported in Table 2.

This question assesses the students’ knowledge of basic arithmetic operators involving integer operands. The student who wrote the first comment praised the question because it had made them aware of the shorthand assignment operator \( % = \). It is also clear from the comment that the student has referred to the coursebook, and provided a page reference for other students, improving the overall quality of the question item.

Table 1. Alternatives and responses.

<table>
<thead>
<tr>
<th>Letter</th>
<th>Answer</th>
<th>Responses (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2</td>
<td>2.26</td>
</tr>
<tr>
<td>B</td>
<td>6</td>
<td>18.55</td>
</tr>
<tr>
<td>C</td>
<td>4</td>
<td>61.65</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
<td>14.29</td>
</tr>
<tr>
<td>E</td>
<td>12</td>
<td>2.26</td>
</tr>
</tbody>
</table>

Table 2. Formative feedback.

Oh good one, I didn’t know what the \( % = \) did. It’s a special shorthand assignment operator, which is on page 26 of the coursebook.

Good question, tests concepts but I personally needed a bit more of an explanation on c--.

I'll look that up now;

Explanation about b\% = 2 could be a bit clearer, and explain that b = b\%2 and say what the % does but im sure that’s up to the person doing it to work it out for themselves, well done.
The second comment also appears to indicate that answering this question has prompted the student to consult the course book (or perhaps some other resource) to find out about the auto-decrement operator.

The third comment suggests that the question could be improved by a better explanation of the shorthand assignment operator, then gives an example of what this improved explanation might look like, and then curiously states that this should be the responsibility of the person answering the question.

5.2. **Example 02**

5.2.1. **Question**
What is the value of d?

```c
int a = 0, b = 1, c = 2;
int d = (!a = = c||b = = c);
```

5.2.2. **Alternatives**
The alternatives created by the author are shown in Table 3, together with the percentage of students who selected each alternative.

5.2.3. **Explanation**
( (!a) = = c) || (b = = c)

!a = !0 = not false = true = 1
(1 = = c) = false = 0
(b = = c) = false = 0
0 || 0 = false or false = false = 0

5.2.4. **Formative feedback and discussion**
The comments created by students who answered the question are reported in Table 4.

Table 3. Alternatives and responses.

<table>
<thead>
<tr>
<th>Letter</th>
<th>Answer</th>
<th>Responses (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-1</td>
<td>4.76</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>33.33</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>52.38</td>
</tr>
<tr>
<td>D</td>
<td>TRUE</td>
<td>0.00</td>
</tr>
<tr>
<td>E</td>
<td>FALSE</td>
<td>9.52</td>
</tr>
</tbody>
</table>
This question assesses whether or not students understand the precedence of the relational and logical operators. The explanation to the question is brief, but helpful, in that it uses parentheses to indicate the correct order of evaluation of the expression.

Four of the five comments listed offer praise to the question author, which we suggested earlier may provide an incentive for them to contribute additional questions. The author of this question did in fact contribute more questions than any other student. At the time we examined the repository, each student was required to have contributed 2 questions, whereas this question author had contributed 27. The question in this example was the fifth one they had written.

The author of the third comment stated that they were not aware of the order of precedence and makes it explicit that the logical "not" operator should be evaluated first. The author of the fourth comment goes a step further and provides a table of the operators and their precedence.

We note that at the time we examined this question, no student had commented on the ambiguity of the boolean values TRUE and FALSE in the alternatives.

6. Discussion

6.1. Learning through observation

Variation theory (Bowden & Marton, 1998; Marton & Booth, 1997) suggests that students learn by experiencing the variation in a phenomenon. One of the benefits provided by constructive evaluation is that students have open access to a wide range of questions, answers and explanations which describe phenomena in different ways.

For example, in a large first-year programming course taught in Java (407 students), we found 114 questions involving the use of loops (Denny et al., 2009b). Each of these questions included an explanation of the correct answer, written from a student perspective. The variation in the
questions, answers and explanations about a specific topic may help students to develop a greater understanding of the topic.

The provision for formative feedback and discussion is an important element of any tool that supports the pedagogy of constructive evaluation, as it provides students with an opportunity to learn from the contributions of other students through vicarious experience (Bandura, 1986). Students learn by viewing a topic in a variety of ways, and by the reinforcement that occurs through the observation of feedback that other students provide on those variations.

6.2. Benefits for instructors

In addition to the direct benefits to students, the process of constructive evaluation produces a large body of test items that have been reviewed and trialled by students. It would take instructors a significant amount of time to produce a databank of questions. The approach described in this article takes very little time or effort from an instructor, since students are responsible for the construction of the questions, the answers, the explanations, the reviews and the discussion. Very little moderation from an instructor is required. This is a low-cost pedagogy that results in a high degree of student engagement and produces learning resources of demonstrable value to students.

Further benefits are provided by the rich source of feedback that instructors can use to reflect on student learning. Hattie (2009) claims that assessments that generate feedback from students to instructors are critical, since a teacher is most effective when they can determine how students are responding to course content. Instructors can easily review the material generated by students to see which topics they are understanding, how they are responding to the questions and areas of course content which are not well represented.

7. Future work

We believe that the pedagogy of constructive evaluation shows great potential. Although we have reported some positive results, we intend to conduct further analysis of student data, including: classifications of the comments students write as feedback; the sophistication of the questions developed relative to the cognitive level they attempt to assess and the application of meta-skills to the various tasks that constitute this pedagogy.

Additional tools that support different styles of questions are in the process of being developed. One of these supports questions that involve free-response answers, and will focus on the peer assessment of both questions and answers. Tools that support MCQs can aggregate student
answers and report the answers succinctly using a histogram. Free-response answers cannot be aggregated, and automated metrics are difficult to generate. Instead, students must compare their own answer with those of others, and apply their critical judgement to determine the quality of their own solution. Although there are numerous tools that support peer assessment reported in the literature (Luxton-Reilly, 2009), none of them is designed specifically to support the peer review of both questions and answers in the structure required for constructive evaluation.

Another tool under development, targeted specifically at introductory programming students, supports questions that require students to develop source code. Using this tool, students will submit a question and a set of unit tests which can be used to provide automated feedback about the correctness of the answers. This tool will support constructive evaluation in the domain of introductory programming code and leverages the advantage of a compiler to provide objective metrics where possible.

We note that the use of web-based tools to support constructive evaluation impose limitations on the domains in which it can be used. It is difficult to easily contribute diagrams and formulae that include mathematical symbols using web-based forms. However, many of the difficulties with diagrams and formulae may be overcome with the use of digital ink, and we are currently investigating ways to support the use of digital ink as an input medium.

8. Conclusions

In this article, we have described constructive evaluation, which is a pedagogy in which students contribute and evaluate questions and answers modeled on traditional assessment items as a shared learning resource. While this pedagogy is not constrained to a particular discipline, we believe it offers benefits that are specifically useful for Computer Science students. In the software industry, practitioners are constantly presented with choices regarding the design and implementation of solutions. Well-informed decisions require eliciting peer feedback, which must be considered and evaluated. The development of meta-skills such as the ability to make critical judgements about the quality of information and the ability to communicate effectively, are fundamentally important and are emphasised by the constructive evaluation pedagogy.

While many possible implementations of constructive evaluation exist, the support of technology is a necessity in all but the smallest classes. We have developed one tool, PeerWise, which focuses on the generation of a repository of MCQs and have evaluated this in the context of first-year Computer Science and Software Engineering courses.

There is a great deal of scope for other tools that support this pedagogy, and instructors in Computer Science are uniquely placed to have a hand in
their design and implementation. We believe that such tools have the potential to engage students in a learning community, focus their attention on course learning outcomes and assessment and promote deep learning.

References


