Thinking Through Computing:
The Power of Learning Communities

Christelle Scharff
Harold Brown
The present paper is reproduced with the generous permission of Taylor & Francis, the publishers of Computer Science Education. It appeared in this journal in 2004 -- Vol. 14, No. 4, Pp. 297-320.

Regarding the posting of this paper on the Website of the Ivan G. Seidenberg School of Computer Science and Information Systems of Pace University:

This is an electronic version of an article published in Computer Science Education (c) 2004 Copyright Taylor & Francis; Computer Science Education is available online at

http://taylorandfrancis.metapress.com/link.asp?id=lpv88muey5r2y440

Christelle Scharff is Assistant Professor of Computer Science at Pace University on the Pace Plaza campus in Manhattan. She holds the Ph.D. in computer science from the Henri Poincaré University of Nancy, in France. She did her doctoral research at LORIA and INRIA Lorraine under the supervision of Dr. Claude Kirchner and Dr. Christopher Lynch.

Dr. Scharff's research interests are in automated deduction and theorem proving, especially the application of automated reasoning to proving the correctness of software and hardware.

Harold Brown is Associate Professor of Philosophy and Religious Studies at Pace University on the New York campus. He holds the Ph.D. from the Graduate Faculty of the New School for Social Research.
Thinking Through Computing: The Power of Learning Communities

Christelle Scharff\textsuperscript{1} and Harold Brown\textsuperscript{2,*}
Pace University, One Pace Plaza, New York, NY 10038, USA, \textsuperscript{1}Department of Computer Science, and \textsuperscript{2}Department of Philosophy

ABSTRACT

A novel approach to improving instruction in an introductory computing course for freshmen non-computer-science majors is to couple it with a logic course as part of a Learning Community. Research has shown that effective Learning Communities can lead to higher academic achievement, better retention rates, diminished faculty isolation, and increased curricular integration. As an import to computer science education, Learning Communities can provide a means for tying together courses to help students better understand the connections between computing courses and other fields, as well across different areas of computing. In this paper, we define Learning Communities and discuss some results from the literature. We present the two courses in our Learning Community, highlight some of the connections we showed between logic and computing, and explain how we assessed the outcomes. We close by describing what we consider to be the most important results from this experience.

1. INTRODUCTION

An interesting pedagogical development in higher education has been the advent of the Learning Community (LC) (Gablenick, MacGregor, Matthews, & Smith, 1990; Matthews, Smith, MacGregor, & Gablenick, 1997; Smith, 2001). Three very good sources for answers to a wide variety of questions on LCs are the guidelines provided in the books by Shapiro and Levine (1999) and Lenning and Ebbers (1999), as well as the National Learning Community

Address correspondence to: Christelle Scharff, Department of Computer Science, Pace University, One Pace Plaza, New York, NY 10038, USA. Tel.: +1-212-346-1016. Fax: +1-212-346-1863. E-mail: cscharff@pace.edu

*E-mail: hbrown@pace.edu

10.1080/0899340042000303456816.00 © Taylor & Francis Ltd.
Project website, dubbed the Learning Community Commons (LCC, 2004). The idea of LCs originated in the 1920s when Alexander Meiklejohn established an experimental college at the University of Wisconsin as a mechanism to prepare students for democratic citizenship. This was to be accomplished through the creation of "community" as well as "a seamless interface between the living and learning environment", that is, between the world outside the University, where students were actually living, and the setting within the University, where students did most of their learning tasks (Gabelnick et al., 1990; Shapiro & Levine, 1999). Since the 1960s, the LC idea has been revived several times to address issues of academic achievement and retention (Smith, 2001). According to the National Learning Community Project, "In higher education, curricular learning communities are classes that are linked or clustered during an academic term, often around an interdisciplinary theme, and enroll a common cohort of students" (LCC, 2004). In this paper, we describe a Learning Community that tied together an introductory course in computing and an introductory logic course.

The strong relationship between mathematical ability and success in introductory computer science courses has been shown in many studies (e.g., Baldwin & Henderson, 2002; Konvalina, Wilman, & Stephens, 1983; Wilson & Shrock, 2001). The importance of this relationship has led to diverse efforts to introduce a stronger mathematical orientation to computer science courses (ACM/IEEE-CS, 2001; Baldwin & Henderson, 2004). One such effort is an on-line repository with content that emphasizes the application of mathematical concepts in the context of computer science and illustrates connections between them (Henderson et al., 2001).

The connection between computer science and logic is as significant as the connection between computer science and mathematics. Courses that frequently make use of and refer to logic concepts include Functional Programming, Formal Languages, Automata Theory, Artificial Intelligence, Databases, and Software Engineering. Examples of connections between logic and computer science include pre- and post-conditions, invariants, recursion, symbolic computation, and formal specification and verification. Many mathematical concepts are also basic concepts in logic. These fundamental concepts are very important for computer science students, but tend to be difficult for novices to learn (Almstrum, 1996).

The opportunity to emphasize the connections between computing and logic suggests the power of including Learning Communities in the computer science curriculum, both as a curricular enhancement and to address issues
important to computer science educators. The body of literature on Learning Communities provides many ideas for import to computer science education research, both in terms of methodologies and evaluation.

In the remainder of this paper, we describe the Learning Community approach used at one university, Pace University, and present it as a model that promotes the goal of showing explicit connections across seemingly unrelated topics. After exploring results from earlier studies involving LCs, we describe the two courses involved in the Thinking Through Computing Learning Community and give examples of connections the students could discover between logic and computing. Next, we explain our assessment approach. Finally, we discuss considerations for future Learning Community offerings and summarize key results.

2. LEARNING COMMUNITIES

2.1. Problems in the Traditional Curricula

According to Gardiner (1998) "society expects college graduates to be able to think critically, solve complex problems, act in a principled manner, be dependable, read, write and speak effectively, have respect for others, be able to adapt to change, and engage in life-long learning". He points out that students generally do not meet expectations, most importantly in three key cognitive areas: "using abstract symbols, epistemology, and principled, ethical reasoning — as well as the ability to work cooperatively in teams with people different from oneself..." (Gardiner, 1998).

It is not likely that the traditional curriculum can lead to the needed changes. We can offer two reasons for this. One reason is the relative isolation of topics that can occur during instruction, which deprives students of important opportunities to develop their abilities. Regardless of how well a discipline relates to other topics, instructors often present content without reference to related disciplines, while students learn as though they are, to use Leibniz' phrase, "windowless monads". This approach leads to difficulties when students need to apply the skills and concepts they have learned, which they will almost certainly need to do while studying a variety of other disciplines. It also has the effect that students do not develop their abilities (Gardiner, 1998).

A second reason the traditional curriculum cannot bring about the needed changes is that education is often seen as merely fact acquisition, a view frequently connected with the traditional lecture. In the paper
“Why universities require computer science students to take math” (Devlin, 2003), Kevin Devlin argues that it is a mistake to view the purpose of education as the acquisition of facts. For one thing, students tend not to remember much of the information learned in a course even a few months later. More importantly, from a positive perspective, Devlin explains “The goal of education is to improve minds, enabling them to acquire abilities and skills to do things they could not do previously.”

2.2. Learning Communities: A Curricular Alternative

Learning Communities have become a growing national movement: more than 400 colleges and universities around the country now offer them, and the number continues to increase (Smith, 2001). One typical mission for LCs is to develop a sense of community (LCC, 2004; Lenning & Ebbers, 1999; Shapiro & Levine, 1999). Other goals and practices are to: (1) incorporate and value diversity, (2) share a culture, (3) foster internal communication, (4) promote caring, trust, and teamwork, (5) involve maintenance processes and governance structures that encourage participation and sharing of leadership tasks, (6) foster the development of young people, and (7) have links with the outside world (Gardner, 1989, cited in Lenning & Ebbers, 1999).

The term learning community also is used in contexts other than that used in this paper. Two other uses are living/learning LCs (LCC, 2004; NSLLC, 2004), and virtual or online LCs (Palloff & Pratt, 1999). Living/learning LCs promote the integration of academic learning and community living and require collaboration between different academic units. For example, resident students who are enrolled in a living/learning LC would live in a common residence hall. Online LCs are possible due to the easy availability of the Internet as a tool for learning, collaborating, and communicating. This means that faculty and students in LCs no longer have to meet face-to-face. An example of an on-line LC is a course where students use the Internet to learn a spoken language by engaging in active exchanges with native speakers.

Lenning and Ebbers (1999) suggest that “two dimensions of learning communities are important for higher education: primary membership and primary form of interaction”. The first dimension, primary membership, “differentiates characteristics the group members have in common”. The types of primary membership that are significant for higher education are the learning organizations of: (1) colleges and universities, (2) faculty learning communities, and (3) student learning communities. The second dimension, primary form of interaction, has three categories that in turn can be used to
characterize any of the three forms of primary membership. These three categories are: physical interaction, where “within-community communication is mostly direct, on site, and in person”; virtual interaction; and correspondent interaction, such as book or fan clubs (Lenning & Ebbers, 1999).

LCs commonly follow one of three models (LCC, 2004):

- the integrated model, where courses are integrated into one interdisciplinary course, or
- the linked course or cluster model, where the curriculum materials in two or more stand-alone courses establish a series of connections between the interwoven courses, or
- the student cohorts/integrative seminar model, where “...a small cohort of students enrolls in larger classes...” An integrative seminar is frequently added to make connections and build community.

Each of these models has advantages and disadvantages, all having to do with the presence or absence of the faculty during class time. Because the integrated model results in just one course, both faculty are generally present for the entire class even if only one is “leading” at a given time. Although in general the linked course model probably requires more work from the faculty and involves greater risks, the possible benefits also are greater (Brown & Salisch, 1996). The LC presented in this paper follows the linked model.

While courses at any level can be included within an LC, lower-level courses, particularly those at the freshman level, have been most frequently chosen to be part of an LC. Most examples of LCs in the scholarly literature involve liberal arts courses: Philosophy with Art, Social Sciences with Literature, English with Psychology. For example, at LaGuardia Community College, an LC called Freedom and Seeing linked Introduction to Philosophy with Introduction to Art, and an LC called Work, Labor and Business in America linked the course Introduction to Social Sciences with the course Work, Labor and Business in American Literature. The Quanta Learning Community project at Daytona Beach Community College offered LCs that tied together three courses each semester. An example of such a team-taught triad of courses was The Quest for Identity: the Search for Identity and Intimacy, which linked English 1 (Composition), Psychology of Adjustment, and Humanities 1.

LCs that follow the linked course model allow the faculty involved to plan their respective curricula so that students will have planned and supervised
opportunities to discover a variety of connections or interrelationships. Developing and elucidating two or three such connections or relationships is generally all that is needed to put to rest the notion of the stand-alone course. Because two — or perhaps three — subjects are involved, the faculty must cooperate to plan the curricula together. As one faculty member develops a particular topic, his or her colleague can refer to, or in some other way make use of, that topic. Thus, not only are connections between the subjects made explicit, but also much of what is learned in each subject turns out to reinforce or help illuminate what is learned in the other.

In this context, beyond the content-related goals of each course, the LC becomes a learning environment that should lead students to as many of the following objectives as possible (Brown & Salisch, 1996; Gabelnick et al., 1990):

- Acquire a deeper understanding of course materials by making connections between courses and disciplines, in part by learning to transfer skills, concepts, and ideas learned in one discipline to appropriate uses in another discipline;
- Learn to find similarities in increasingly disparate subject areas in order to become better able to formulate syntheses that might, for example, aid in the solution of a broader range of problems;
- Import and export knowledge from one course to another;
- Experience increased interaction with other students and faculty, as well as a rich learning-centered community inside and outside of the classroom;
- Participate in active and collaborative learning; and
- Explore and begin to understand diverse perspectives.

LCs appear to provide solutions for many of the expectation issues raised by Gardiner (1998) and seem to create the “inviting, intellectually stimulating, ethically principled, and emotionally supportive environment [that] can enhance students’ academic achievement, and their willingness to remain in college” (Gardiner, 1998). Indeed, LCs provide many benefits for students: increased academic achievement, better retention rates, greater satisfaction with college life, reduced time to degree, improved quality of thinking and communicating, better understanding of self and others, and greater ability to bridge the gap between the academic and social worlds. In addition, there are benefits for faculty: diminished isolation, shared purpose and cooperation among faculty colleagues, increased curricular integration, a fresh approach to one’s discipline, and
increased satisfaction with their students' learning (Lenning & Ebbers, 1999; Palloff & Pratt, 1999).

2.3. Learning Communities at Pace University
At Pace University the primary obligation to develop and teach courses that are part of the general education of all students, referred to as "the core curriculum", falls to the faculty of the Dyson College of the Arts and Sciences. In 1998 the Dean of Dyson College appointed a Task Force made up primarily of faculty of the Dyson College, but including representatives from the other schools of the University, to convene for the purpose of reviewing the old core and possibly making recommendations regarding a new core. The review by the Task Force concluded that the old core merely distributed the required number of courses among a variety of disciplines with no structure built into the core to help students see the connections between otherwise disparate subjects.

An innovation in the new core was the requirement that all new students had to complete at least one Learning Community. The Freshman Thinking Project (FTP), founded and directed at Pace University by the philosopher co-author of this paper, was the precursor of the LC pilot. FTP featured cohorts of students registering for clusters of three linked classes in both semesters of their first year. From the beginning of FTP in 1986 until it was terminated in 1992 for budgetary reasons, the students reported in their evaluations that the most important feature of the programme\(^1\) was the sense of community it generated (Brown & Salisch, 1996).

The LC requirement of the new core led to an unprecedented effort to recruit faculty from every school of the University to implement the LC pilot. Twelve LCs were offered in the initial semester, Fall 2002. From the initial LC pilot to the present, only the integrated and linked LC course models have been used (LCC, 2004). Some of those pairings were:

- English with Ethics (e.g., Writing Right(s): Ethical Dilemmas in Human Relationships);
- Literature with Political Science (e.g., American Slavery: Politics and Literature);

---

\(^1\)To distinguish between the two distinct meanings of the word program, we have adopted the convention of using programme for the curricular meaning and program to refer to a computer program.
• English with Biology (e.g., Perspectives on Human Biology and Contemporary Society);
• Religious Studies and History (e.g., The Sacred and the Secular in East Asia);
• Italian with Acting (e.g., La Dolce Vita);
• Mathematics with Astronomy (e.g., A Mathematical Journey with Astronomy);
• English with Law (e.g., The Longest War: Violence against Women); and
• English with Computing (e.g., The Written Word in the Internet Age).

The pairing that is the focus of this paper, Computing and Philosophy, coupled an introductory computing course for non-computer-science majors with a logic course offered by the Philosophy Department. The title of this LC, Thinking Through Computing, reflected our belief that each discipline supports the other. While the linked course model that we used for our LC has been used many times since Fall 2002, we have not yet repeated the pairing of logic and computing because of scheduling issues.

3. THE THINKING THROUGH COMPUTING LEARNING COMMUNITY

This section outlines the topics of the logic and computing courses involved in the Thinking Through Computing LC. The syllabus for each course, as well as supplemental materials such as sample E-portfolios and a guide for writing essays, are at the URL: www.csis.pace.edu/~scharff/learningcommunity/

3.1. Computer and Information Systems Course
The Computer Information Systems course CIS 101 is a foundation core course required for all Pace University undergraduates who are not matriculated in the School of Computer Science and Information Systems. During the two hour weekly lecture, students acquire conceptual understanding of fundamental concepts and operational skill using guided hands-on exercises in the basics of computer hardware and software. The two hour weekly lab sessions include use of common application packages and introductory procedural programming during two hour lab sessions every week. The course is organized around the following topics: problem solving, introduction to algorithms, introduction to programming in Visual Basic,
computer organization, data representation, networking, spreadsheets using Excel, designing web pages in HTML, and ethical issues in computing.

3.2. Philosophy Course
The Philosophy course, Philosophy 253: Logic, is required for all students majoring in Philosophy and Religious Studies and is recommended for pre-law students as well as psychology and business majors. Primarily devoted to propositional logic, this course helps students improve skills related to identifying arguments, uncovering assumptions, and solving problems. The course begins with definitions and brief descriptions of basic concepts such as argument, deduction, induction, validity, and soundness, as well as some important theories of truth and meaning. The course continues with definitions of the five truth-functional connectives, with students learning to translate sentences and then arguments from English into symbols. Students test translated arguments for validity with truth tables or prove arguments valid with formal proofs using a set of inference rules based on a natural deductive system. Toward the end of the semester students learn to identify and analyze various kinds of arguments in natural language, including arguments from propositional logic, syllogisms, and fallacies.

3.3. A Framework to Emphasize Connections
Connections are pointers between related topics. In the Thinking Through Computing LC, which featured the two stand-alone courses described above, connections were intended to lead students from one of the courses to the other and back again. The infrastructure of the LC made it possible to import and export knowledge from one course to the other in "real time". As professors of computer science and philosophy, we collaborated in using our training and experience to create a combined syllabus in which the elements of the typical syllabi for these courses were reordered and contextualized to maximize connections between the disciplines. The crucial factor in our curricular decisions was always whether a change would enhance instruction. Each of us often attended and participated in the other's lectures to reinforce the notion of connectedness, increase the interaction among students and faculty, and create an enriched learning-centered environment. In this section and the next we describe the connections we emphasized in this LC and show how connections were strengthened and broadened with common assignments.

A fundamental connection between the two courses was the link between formal proofs on the one hand, and algorithms and programming on the other.
Table 1. Examples of Inference Rules, Logical Symbols, and an Argument (Pospese, 2000).

<table>
<thead>
<tr>
<th>Key to symbols used in this table and other parts of the article</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp; Symbol for the conjunctive connective</td>
</tr>
<tr>
<td>→ Symbol for the conditional connective</td>
</tr>
<tr>
<td>~ Symbol for the negation connective</td>
</tr>
<tr>
<td>∨ Symbol for the disjunctive connective</td>
</tr>
<tr>
<td>→ Symbol for the biconditional connective</td>
</tr>
<tr>
<td>├ Symbol indicating that what follows is the conclusion</td>
</tr>
</tbody>
</table>

Inference rules

- Arrow Out [→O]: A → B, A ├ B
- And Out [&O]: A & B ├ A
- And In [&I]: A, B ├ A & B
- (A & A) → B, A ├ B

In logic, students learned the following definition: "A formal proof for any deductive argument is a series of statements such that every statement in the series is either an assumption of that argument or follows from one or more previous statements by a valid rule of inference."

A useful example in this context comes from Howard Pospesel's text (Pospesel, 2000). At the point when we introduced the argument given at the end of Table 1, students were using the inference rules shown in the middle of Table 1. Several aspects of constructing a proof for the argument (A & A) → B, A ├ B illustrate points important for students studying computer science as well as logic:

- **The need for careful, rigorous adherence to the form of the inference rules.**
  Students attempt to take steps such as inferring A from (A & A) → B. Their thinking seems to be that because A is included in (A & A) one should be able to deduce A, even though none of the inference rules permits inference of the antecedent (or any part of the antecedent) of a conditional.

- **Developing strategies for proofs.** Analogous to other academic processes, such as writing essays or solving problems, students need to be able to see the larger picture. For a proof that means considering the conclusion of the argument in light of the given premises. Because proofs for valid deductive arguments can be completed in a finite (generally small) number of steps, students should be encouraged to begin working out a strategy by thinking about which inference rule might be used to derive B from assumptions (A & A) → B and A.
Table 2. Two Proofs of \((A \land A) \rightarrow B, A \vdash B\).

<table>
<thead>
<tr>
<th>Proof 1 (liberal interpretation of &amp;\lor)</th>
<th>Proof 2 (step-by-step)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) ((A \land A) \rightarrow B)</td>
<td>(1) ((A \land A) \rightarrow B)</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>(2) A</td>
<td>(2) A</td>
</tr>
<tr>
<td>(3) ((A \land A) \lor 2, 2 &amp; I)</td>
<td>(3) ({(A \land A) \rightarrow B} \lor A)</td>
</tr>
<tr>
<td>(4) B</td>
<td>(4) A</td>
</tr>
<tr>
<td></td>
<td>3 &amp; O</td>
</tr>
<tr>
<td></td>
<td>(5) A &amp; A</td>
</tr>
<tr>
<td></td>
<td>2, 4 &amp; I</td>
</tr>
<tr>
<td></td>
<td>(6) B</td>
</tr>
<tr>
<td></td>
<td>1, 5 \rightarrow O</td>
</tr>
</tbody>
</table>

- Only one inference on each line of the proof. This guideline is critical and critically important for designing algorithms and a fortiori for programming, a program being a sequence of instructions. Pospesel (2000, p. 38) offers a 'proof' that uses what he describes as a 'liberal' interpretation of the &\lor rule, that is, several steps are combined and assigned to the application of the &\lor rule. Our objection to this interpretation of the &\lor rule is that students then might believe that they are free from the restriction that only one inference is allowed in each step of a proof. Table 2 shows the proof of the argument \((A \land A) \rightarrow B, A \vdash B\) using the 'liberal' interpretation of the &\lor rule (Proof 1) and using a step-by-step proof (Proof 2).

Another connection fundamental to the LC was the link between conditional statements in the two areas. In logic, this refers to truth-functional compound statements and is represented using the symbol \(\rightarrow\). In the computing course, this concerns the if-then[else] conditional statements of programming languages. Students transformed natural language sentences into propositions and then into the equivalent Visual Basic conditional statements. They also did the reverse, that is, transforming Visual Basic conditional statements into propositions. For example, consider the sentence: "If it is hot, I will drink lemonade, otherwise, I will drink soup." This sentence can be translated into the proposition: \(h \rightarrow l \& \sim h \rightarrow s\), and then into the Visual Basic statement:

```vbnet
If txtCondition.Text = "Hot" Then
    TxtStatement.Text = "I will drink lemonade"
Else
    TxtStatement.Text = "I will drink soup"
End If
```

Finally, students were led to discover logic as part of everyday life. For example, they saw logic being used in application packages such as Excel as
well as in search engines when they surfed the web. Students used Excel to create truth tables and applied their knowledge in logic to transform advanced search queries of Google into propositions. Indeed, advanced search in Google offers the possibility of finding web pages containing ‘All the words’, ‘At least one’ or ‘None of the words’ of a sequence of words of a search. Students came to understand that logic underlies all of these concepts.

3.4. Strengthening the Connections With Assignments
Giving a variety of joint assignments assisted us in strengthening and broadening the connections between concepts in the two courses. The three common assignments included a programming task, an essay-writing task, and the design of an E-portfolio (Hawisher & Selfe, 1997). The E-portfolio was used as an assessment tool to enhance students’ learning as discussed in the next section. Because these courses were part of a Writing Enhanced (WE) Course programme that encourages writing across the curriculum, we included a substantial amount of writing in both courses. Faculty teaching WE courses were encouraged to provide students with a writing guide so that their expectations were clear. Another feature of the WE programme was the use of multiple drafts for writing assignments. Students were encouraged to work at improving their writing through submission of first drafts of an assignment. These first drafts were then returned to the student with comments and suggestions for improving the final version of the assignment. The same approach was used for the submission of the programming assignment.

3.4.1. Programming Assignment
The first common assignment combined truth tables from the logic course, programming in Visual Basic from the computing course, and a writing assignment. Very early in the semester students learned the connectives: ‘→’, ‘&’, ‘∨’, ‘←→’, ‘¬’. Having learned in the logic class to translate various truth-functional compound statements from English into symbols, students then learned to test the validity of deductive arguments using truth tables. The students were given a programming assignment to design a Visual Basic application called “Learn to do truth tables” that would determine whether the user could construct truth tables for the logical connectives: ‘→’, ‘&’, ‘∨’, and ‘←→’ (the program required primarily if-then-else conditional statements). Students also had to write a user manual explaining how to use the application.
The connection between the logic and computing classes that was emphasized in this assignment was the use of truth tables and the if-then-else conditional statements present in programming languages. The targeted outcomes were problem analyses (e.g., via the understanding of the provided requirements of the application leading to a design), problem solving (e.g., via the design of the solution to the problem and its implementation in Visual Basic), communication (e.g., via writing a user manual and designing an appropriate graphical user interface for the application), and technological fluency (e.g., via the use of diverse applications running under Windows). Students had the opportunity to discuss the programming assignment with the instructors while they were working in the labs. They submitted the assignment for grading after receiving feedback on their first drafts.

3.4.2. Essay Writing
The second common assignment was an essay where students had to identify a social or ethical problem that has resulted from technological innovation, describe the problem in detail, and explore a possible solution. Ethical issues in computing was a required topic of the computing course, so it was natural to include the topic of ethical issues as part of a lecture that was team-taught by the two instructors. This assignment addressed communication, research, and technological fluency. Specific writing and grading criteria for the essay were given to the students as part of the specifications of the assignment. Students submitted first drafts, which were returned with comments and suggestions for revision before they submitted their final drafts. While every aspect of these essays was considered, the most important consideration in grading this assignment was the way students developed their arguments.

3.4.3. E-portfolio
The third common assignment engaged students in creating an E-portfolio to collect, reflect on, and showcase the work they accomplished in the Learning Community. The E-portfolio was also used as an assessment tool. Students used HTML to design a very personal E-portfolio containing:

1. a brief autobiographical presentation;
2. a presentation about the learning community;
3. links to useful web pages;
4. reflective logs where the student elaborated on the connections between the two courses and gave their impressions of the learning community; and
5. links to the students' work, including:
   a. the Visual Basic application "Learn to do truth tables" and its user manual;
   b. a review of the movie "Pirates of Silicon Valley'';
   c. a review of the Broadway show "Proof", which was attended by the entire class; and
   d. a link to their essay on ethical issues in computing.

The students had the opportunity to make a presentation of their E-portfolios to the whole class. This assignment addressed communication, information literacy, research, and technological fluency. Examples of E-portfolios are available at www.csis.pace.edu/~scharff/learningcommunity/

4. ASSESSMENT AND RESULTS

Our assessment plan reflects the importance we attached to the assessment process. We designed the plan to document the results we anticipated from our LC. We believe that this assessment plan will be useful in other contexts as well.

The LC classes were originally composed of 13 non-computer-science majors: 8 business majors, 1 political science major, 1 psychology major, and 3 students who had not yet declared a major. By the middle of the semester, one of the students who had not yet declared a major had dropped out of the logic course, so that only 12 students finished logic while 13 completed the computing course. Since the students responded to the assessment instruments for our LC during the logic class, there were only 12 responses to the LC perceptions survey (described below). While such a small sample does not permit us to predict any general trends, the results do demonstrate the success of this approach. We also plan to use the results as the basis for a future offering of the Thinking Through Computing LC as well as other LCs that each of us might engage in with other partners.

4.1. Assessment Plan

The main goal of our assessment plan was to evaluate the impact of this LC infrastructure. There were two main aspects to this impact: as a means of

---

2The student who did not complete the logic class completed the Visual Basic programming assignment that was the first assignment, but did not complete the other linked assignments. The student did not receive credit for the Learning Community requirement of the core, but did receive credit for the computing course.
emphasizing connections between logic and computing, and for facilitating import and export of knowledge between logic and computing in “real time”. We focused on the following questions:

- Did students recognize the connections between logic and computing?
- How did students perceive the LC?
- Did the students develop particular interests in one or both of the subjects taught in this LC?

Our assessment plan was based on formative and summative assessment. Formative assessment is defined as feedback to students “in ways that enable the student to learn better, or when students can engage in a similar, self-reflective process” (Black & William, 1998a, 1998b; FTE, 1999). Formative assessment was implemented by offering students the opportunity to obtain feedback on their assignments; they could revise their work after receiving feedback and before the assignment was graded. E-portfolios were also used as formative assessment tools (Anderson, Feather-Ganon, & Scharff, 2003): students elaborated on the connections between logic and computing and expressed their opinions about the LC in reflective logs integrated into their E-portfolios.

Summative assessment is defined as “the attempt to summarize student learning at some point in time, say the end of a course” (Black & William, 1998a, 1998b; FTE, 1999). To assess change in student performance we administered the Cornell Critical Thinking Test Level X (Ennis & Millman, 1985) at the beginning and at the end of the semester, then studied and compared the results on the portions dedicated to induction and deduction. In addition we administered at mid-semester and end-semester an 11-item perceptions survey designed for use in all LC courses by the Core Task Force Sub-committee on Assessment of Pace University. The survey questioned students about their understanding of the connections between the two courses and their impressions of the LC.

During the semester, we used the formative assessment results to adapt and enhance our teaching. Although the formative assessment was explicitly directed at the students, we believe that the success of our formative assessment efforts influenced the positive changes we report below in our summative assessment results. The feedback we gave our students helped them to improve their work, which led to greater student satisfaction and more positive feedback. Unfortunately we did not recognize the interplay between formative and summative assessment sufficiently early to include in our student evaluation form a question specifically directed at finding out the effect that students felt the formative assessment had on their course work.
4.2. Student Feedback
The students used the reflective log of their E-portfolios to elaborate on their impressions of the LC and to describe the connections they discovered and understood between logic and computing. They cited Boolean expressions, truth tables, if-then statements, and problem solving versus proof construction as connections between computing and logic. In fact, every connection that we emphasized between the two courses was cited in the logs at least once. Below are some samples of the reflections written by these students.

This learning experience gave me a broader understanding of computer programs and logic concepts. It allowed me to apply the logic concepts I learned to solve problems involving computers. Knowing basic logic concepts made the computer science part of the course easier.

My experience with the learning community was quite helpful. With this experience I realized how different subjects can relate to each other. In this course both professors were present and it was helpful for both professors to be present because they interacted with both subjects. In this course I had the opportunity to work with other students. Together we came up with different ideas on how these two classes relate with one another.

In other courses there is nothing to compare your learning with. In this LC, we can say: “I remember that from the logic class.”

In order to compare the connections noted by the LC students, we decided to survey computer science students from another setting. Intuitively, we thought that advanced computer science majors should be aware of connections between logic and computing. As an extra-credit assignment, the computer science instructor co-author of this paper asked the junior and senior students enrolled in her advanced course for computer science majors to describe one or two connections they saw between computer science and logic. Only 7 of the 15 students provided acceptable answers. These responses included roughly the same ideas given by the freshmen students enrolled in the LC, including: truth tables, digital circuits, conditional statements, and manipulation of abstract concepts.

4.3. Surveys and Results
We administered the 11-item perceptions survey in Table 3 twice during class time in the logic course, once in the middle and once at the end of the semester. Students answered each question using the scale: Strongly
Table 3. Results of "Strongly Agree" and "Agree" from LC Perceptions Survey.

<table>
<thead>
<tr>
<th>Survey questions</th>
<th>Thinking Through Computing LC</th>
<th>All LCs at Pace U. – Fall 2002</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mid-semester (N = 12)</td>
<td>End of semester (N = 12)</td>
</tr>
<tr>
<td></td>
<td>End of semester (N in parentheses)</td>
<td></td>
</tr>
<tr>
<td>1. Did you recognize connections between the two disciplines or courses?</td>
<td>41%</td>
<td>83%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>95% (N = 112)</td>
</tr>
<tr>
<td>2. Did the faculty members organize joint assignments to facilitate learning?</td>
<td>41%</td>
<td>91%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>91% (N = 113)</td>
</tr>
<tr>
<td>3. Did assignments reflect the common themes?</td>
<td>83%</td>
<td>91%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>97% (N = 111)</td>
</tr>
<tr>
<td>4. Were the joint events well planned and did they increase your understanding of the course material?</td>
<td>83%</td>
<td>91%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>85% (N = 113)</td>
</tr>
<tr>
<td>5. Did looking at common themes from different perspectives increase your learning experience environment?</td>
<td>75%</td>
<td>75%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>94% (N = 104)</td>
</tr>
<tr>
<td>6. Did working collaboratively with your classmates in the LC create a positive learning experience environment?</td>
<td>83%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>96% (N = 113)</td>
</tr>
<tr>
<td>7. Did you feel comfortable visiting and/or communicating with your professors outside of class?</td>
<td>75%</td>
<td>83%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>96% (N = 113)</td>
</tr>
<tr>
<td>8. Did you form one or more friendships in the LC?</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>97% (N = 113)</td>
</tr>
<tr>
<td>9. Would you register for another LC?</td>
<td>75%</td>
<td>75%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>87% (N = 96)</td>
</tr>
<tr>
<td>10. Would you recommend this LC to a friend?</td>
<td>75%</td>
<td>75%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>89% (N = 96)</td>
</tr>
<tr>
<td>11. Would you take another class in one or both of the subjects taught in this LC?</td>
<td>66%</td>
<td>66%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84% (N = 95)</td>
</tr>
</tbody>
</table>

Agree/Agree/Neutral/Disagree/Strongly Disagree. The 12 students who completed both courses in the LC responded to both rounds of this optional survey.

For comparison, the last column of Table 3 shows the compiled results from the end of the semester of all 12 LCs that were offered during Fall 2002 at Pace University. The twelve LCs included a total of 163 student enrollments. Only the results of the end-semester surveys of the twelve LCs are provided because the mid-semester survey results were only available to instructors to allow
them to adapt and enhance their teaching. The number of students who responded to each question varies, so this number is provided in parentheses.

The results of the mid-semester and end-semester surveys show improved perceptions with respect to questions concerned with the connections between the two courses (questions 1, 2, 3, and 4). An important result is that by the end of the semester 83% of the students (i.e., 10 of the 12 who responded) reported that they could recognize the connections between computing and logic, whereas on the mid-semester survey, only 41% of the students (i.e., 5 of the 12 who responded) reported that they saw these connections. Students also reported that this LC improved collaboration among students, as well as students' communication with the professors. There were no changes in the reported desire to register for another LC or to recommend this LC to a friend. After talking with students, including some conversations after the end of the semester, we concluded that this lack of change was probably due to students' perceptions that our LC required lots of work and was very time consuming. We believe that the differences in the results between our LC and the other LCs were based on the fact that our LC was the only relatively "technical" LC; no other LC linked two disciplines comparable to Logic and Computing. Overall, the results show that students at Pace University supported the Learning Community concept in general, and that students in our LC supported our Learning Community.

4.4. Cornell Critical Thinking Test
We administered the Cornell Critical Thinking Test Level X (Ennis & Millman, 1985) twice, at the beginning and at the end of the semester. According to this test, critical thinking is described in terms of four major abilities: inductive thinking, judging credibility of observation reports, deductive thinking, and assumption identification. We studied and compared the results on the parts of the test most related to the courses we taught: induction and deduction. The test is composed of 76 questions, including 23 questions on induction and 14 questions on deduction.

Twelve students took both rounds of this test during class time in the logic course. We found no significant improvements in the results of the tests with respect to the mean of the total score. Only three students (25%) did better on the induction and deduction questions at the end of the semester. In the future, we hope to find a test that better suits our needs. Such a test would focus on deduction, the primary focus of the logic course, rather than induction, a fifteen-minute topic during one lecture.
5. FUTURE DIRECTIONS

We do plan to offer another LC that pairs logic and computing in the future, although the timing is not yet determined. Research questions we will consider in future offerings of this LC pairing include the following:

- Did the emphasis on connections between logic and computing help students in programming?
- Did the emphasis on connections between logic and computing foster better performance in the logic class?
- Did teaching computing and logic with explicit attention to the connections achieve more than teaching these topics separately?

The strongest evidence for answering these questions would result from an experimental design using control groups to provide the basis for comparison. Because this LC was a pilot and there was no control group for comparison, we were unable to fully investigate these questions. Even though we could have accessed grades of students who had taken non-LC logic courses during previous semesters, we did not compare their grades with those of the students in the LC logic course. The reason is that the samples were drawn from different populations. In the LC course, all students were freshmen, while students who took the non-LC logic courses were at least sophomores. In addition, finding a control group for the computing class would have been difficult, because each section of CIS 101 is taught by a different instructor and each instructor uses different technologies and therefore gives different assignments.

The implications of bringing a course in computing and a course on logic together into a Learning Community go beyond the short-range impact that the LC had on the participating students. The improvement in critical thinking abilities should help students in a wide range of courses. We did not obtain significant results regarding improvements in critical-thinking abilities, so a goal for our next offering of this LC is to do more to improve both our delivery and assessment of critical thinking. We expected that because of the mutual reinforcement between the students' work with formal proofs in logic and programming in computing, students in our LC would demonstrate improved abilities to construct formal proofs and programs.

Because the Thinking through Computing LC was part of a Writing Enhanced (WE) Course programme that encourages writing across the curriculum, the LC included a substantial amount of writing. It would be interesting to investigate how writing helped students learn the concepts of the
courses of the LC, since that result was not developed in this pilot offering. The extensive literature on Writing Across the Curriculum offers ideas for how to approach this; for example, Fulwiler (2000) suggests several approaches to evaluating student learning within programmes that focus on “writing as a mode of learning”.

6. FINAL THOUGHTS AND IDEAS

Results from the LC described in this paper, other LCs offered at Pace University, and LCs offered at other universities suggest that Learning Communities offer computer science education many potential improvements to current teaching approaches. In particular, LCs can influence faculty as they develop collaborative interdisciplinary teaching and create innovative LC-related activities in and out of the classroom. Furthermore, LCs may enable faculty to better assess student learning, provide mutual support for LC courses, and improve programme recruitment. We discuss these points in the remainder of this section.

LCs can have a significant impact on faculty (Lenning & Ebbers, 1999). When they realize the amount of work that is involved in developing a Learning Community, it can be difficult to convince faculty to commit to developing new LCs. However, faculty who have a successful experience teaching an LC become lifelong adherents of this pedagogy and report that it energizes their teaching in other courses as well (Brown & Salisch, 1996). This particular LC involved a senior professor with over 30 years of experience and a junior assistant professor. The combination worked very well. The senior professor was able to learn a great deal about computing. The nature of the LC allowed the senior professor to assume a very natural role in the computing classes, including some of the lab sessions. For the junior faculty member, this meant that issues sometimes associated with the presence of a peer observer did not apply. This led to comfortable and helpful exchanges about teaching and allowed the instructors to incorporate a variety of LC-related activities in and out the classroom. One example was a discussion that was held after a lecture on ethical issues in computing that was team-taught by the two instructors. Students said they found it interesting to explore different views of ethical issues in computing; experiences such as this can be particularly valuable because students tend to “view the world in rigid categories of black-white, right-wrong, and good-bad…” (Gardiner, 1998).
An unusual but exciting outside activity was to attend the Broadway show *Proof*; this experience provided students with a unique opportunity to connect the topics of the LC with the "real world". Each student included a review of the show in his or her E-portfolio.

A difficult issue for assessment under any circumstance is determining whether students have learned material well enough to use it in "real life". An LC that combines a computing course with a course in another area, for example business, would give computing professors insights into whether their students had learned a skill or concept well enough to apply it in "real life" situations in the other course. The computing and business faculty could create business cases for which the solution required computing skills, concepts, and practices; the computing course could include instruction designed to help students discover strategies that could lead to solutions in the business course. Other interdisciplinary examples of possible pairings are to link a liberal arts course with a computer science course, such as propositional or predicate logic with a first or second programming course, technical writing with software engineering, digital art with web-design, or psychology or marketing with human and computer interaction.

Most examples of LCs in the scholarly literature involve liberal arts courses; the same trend has held at Pace University. Perhaps more liberal arts faculty are willing to teach in LCs due to the guarantee of student enrollments: because all students must take some liberal arts courses at virtually all American colleges and universities, it may be easier to find students to register for these LCs. The question is how to establish a greater number of LCs that relate to technical areas such as computing. Such LCs in computing programmes could be used both to connect computing courses to other computing courses and to connect computing courses with courses in seemingly unrelated disciplines and topics. For example, LCs could combine any of a variety of computing courses with courses in business or engineering. Such LCs have implications for possible import into computer science education research beyond what we were able to realize in the *Thinking Through Computing* LC. Researchers and planners interested in implementing an LC should consult the extensive literature on LCs for guidance in a variety of choices. For example, the planner must begin by choosing an appropriate LC model: the integrated model, the cluster model, or the student cohorts/integrative seminar model (LCC, 2004).

LCs within the computer science curriculum could allow computer science courses to provide mutual support for one another. Even when there are prerequisites, computer science topics are often taught as if they are relatively
independent of one another. Ideas for linking computer science courses include discrete mathematics with data structures, algorithms and data structures with databases, databases with software engineering, and data structures with operating systems. As an example of a pairing at a more advanced stage of studies, consider a typical databases course. Such a course normally covers data manipulation operations, normalization, query optimization, and database management system optimizations, all of which refer to and use advanced notions of algorithms and data structures; this suggests packaging the three courses databases, advanced algorithms, and data structures.

Introducing LCs in the curriculum can serve as a tool for programme recruitment. Faculty wishing to attract good students to their disciplines must offer courses that will attract those students. A well thought out LC with an attractive title can have that effect. Students intimidated when faced with studying a difficult subject might find that subject less frightening if it were linked with another subject that they perceive as more accessible. At present, interdisciplinary pairings appear to be particularly interesting and important in attracting students to computer science, as many students apparently pursue computer science because of the influence of computers in other fields. In addition, the interdisciplinary approach might help attract women in computer science. De Palma (2001) showed that women are more attracted to mathematics than computer science and offered suggestions for teaching computer science like mathematics. LCs linking a mathematics course with a computer science course could promote this idea by integrating the two topics.

In conclusion, Learning Communities offer many ideas for import to computer science education research. Because LCs are not yet common in computing programmes, faculty planners face challenges in developing LCs. Most challenging is the careful reading of the literature and the years of experimentation that are necessary to select the LC models, methodologies, and evaluation techniques most appropriate for computing classes and programmes. Continued exploration with the idea of the Learning Community and incorporation of results from the literature offers great promise for computer science education.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers of this paper, the guest editors of this Special Issue of the Journal of Computer Science Education for their useful comments, and the
Core Task Force Sub-committee on Assessment at Pace University for their help in providing survey results for this paper.

REFERENCES


The Ivan G. Seidenberg  
School of Computer Science and Information Systems  
Pace University  
Technical Report Series

EDITORIAL BOARD

Editor:
Allen Stix, Computer Science, Pace--Westchester

Associate Editors:
Constance A. Knapp, Information Systems, Pace--Westchester
Susan M. Merritt, Dean, CSIS--Pace

Members:
Howard S. Blum, Computer Science, Pace--New York
Mary F. Courtney, Computer Science, Pace--Westchester
Nicholas J. DeLillo, Mathematics and Computer Science, Manhattan College
Fred Grossman, Information Systems; Doctor of Professional Studies, Pace--New York and White Plains
Fran Goertzel Gustavson, Information Systems, Pace--Westchester
Joseph F. Malerba, Computer Science, Pace--Westchester
John S. Malolzzi, Computer Information Sciences, Iona College
John C. Molluzzo, Information Systems, Pace--New York
Pauline Mosley, Technology Systems, Pace--New York
Narayan S. Murthy, Computer Science, Pace--New York
Catherine Ricardo, Computer Information Sciences, Iona College
Judith E. Sullivan, CSIS Alumna, MS in CS from Pace--Westchester
Sylvester Tuohy, Computer Science, Pace--Westchester

The School of Computer Science and Information Systems, through the Technical Report Series, provides members of the community an opportunity to disseminate the results of their research by publishing monographs, working papers, and tutorials. Technical Reports is a place where scholarly striving is respected.

All preprints and recent reprints are requested and accepted. New manuscripts are read by two members of the editorial board; the editor decides upon publication. Authors, please note that production is Xerographic from your submission. Statements of policy and mission may be found in issues #29 (April 1990) and #34 (September 1990).

Please direct submissions as well as requests for single copies to:

Allen Stix  
The Ivan G. Seidenberg School of Computer Science and Information Systems  
Goldstein Academic Center  
Pace University  
861 Bedford Road  
Pleasantville, NY 10570-2799