Applying Abstraction to Master Complexity:
The Comparison of Abstraction Ability in Computer Science Majors
with Students in Other Disciplines

Jonathan H. Hill
Bernice J. Houle
Susan M. Merritt
Allen Stix
The current paper was presented on May 11, 2008 at the
Second Workshop on the Role of Abstraction in Software Engineering
which was part of the
30th International Conference on Software Engineering (ICSE)
held in Leipzig, Germany.

Jonathan H. Hill is the Director of Special Programs and Projects and Assistant Dean in the Seidenberg School of Computer Science and Information Systems at Pace University. In addition, he is Assistant Professor in the Technology Systems Department. The present paper is derived from the investigation forming the basis of his doctoral dissertation.

Dr. Hill holds a Doctorate of Professional Studies in Computer Studies from the Seidenberg School. His research interests are in: (1) the nature of abstract thinking as it relates to success in designing software, success in computing majors, and attraction toward studying computing as well as in (2) e-Commerce. Dr. Hill came to Pace in 2003 from Travelocity.com, holding an MBA from the Zicklin School of Business - Baruch College, City University of New York.

Bernice J. Houle is Associate Dean in the Seidenberg School of Computer Science and Information Systems at Pace University. She earned a BS in mathematics and an MS in education from Pace University and holds the Ph.D. from New York University.

Susan M. Merritt served for 25 years as the first Dean of the Seidenberg School from its inception in 1983 through June 2008. She continues to play an important, influential role in her position as Professor of Computer Science. Professor Merritt holds a baccalaureate in mathematics from the Catholic University of America and an MS and Ph.D. in computer science from the Courant Institute of Mathematical Science at New York University. Her research interests are in algorithm design and analysis, programming languages (in particular, transformational programming), computing and social responsibility, and on the impacts of computers in education and culture in general.

Allen Stix is an associate professor in the Computer Science Department.
Applying Abstraction to Master Complexity: 
The Comparison of Abstraction Ability in Computer Science Majors with Students in Other Disciplines

Jonathan H. Hill, D.P.S. 
Seidenberg School of CSIS 
Pace University, NY, USA 
1 Pace Plaza 
New York, NY USA 10038 
44 212 346 1864 
jhill@pace.edu

Bernice J. Houle, Ph.D. 
Seidenberg School of CSIS 
Pace University, NY, USA 
861 Bedford Road 
Pleasantville, NY USA 10570, 
44 914 773 3592 
bhoule@pace.edu

Susan M. Merritt, Ph.D. 
Seidenberg School of CSIS 
Pace University, NY USA 
861 Bedford Road 
Pleasantville, NY USA 10570 
44 914 773 3750 
smerritt@pace.edu

Allen Stix, Ph.D. 
Seidenberg School of CSIS 
Pace University, NY, USA 
861 Bedford Road 
Pleasantville, NY USA 10570 
44 914 773 3833 
astix@pace.edu

ABSTRACT
Aptitude for managing abstraction may be a distinguishing characteristic of computer science majors. If this is so, and if this aptitude can be recognized among potential majors, those who are well suited for computer science but have not considered it a major can be made aware of the possibility. Abstraction, as a human ability, is comprised of two complementary aspects: clearing away details to build simplifications and deriving generalizations that illuminate essentials. Agreement exists that this ability may be nurtured through instruction and experience, but that it rests upon a natural aptitude that is possessed by few. Agreement exists that this natural aptitude is assessable, although no instrument yet exists for measuring it efficiently among prospective computer science majors who have not begun computer science coursework. This paper deals with a study done at New York’s Pace University to test undergraduate students across a range of majors for abstraction ability.

Categories and Subject Descriptors
D.2. [Software Engineering]: General.
K.3.2 [Computers and Education] Computer and Information Science Education

General Terms
Management, Measurement, Human Factors,

Keywords
Abstraction, testing, cognition, computer science education, discipline

1. INTRODUCTION
The context of, and prime motivation for, this study is the decline in the number of students electing to major in computer science. This decline has been taking place over the past five years. It is well documented [1, 7] and a source of concern to academia, industry and government. If there is a distinguishing aptitude characteristic of the discipline, and if this aptitude could be recognized among new freshmen, perhaps students with potential, who may not have considered majoring in computer science, could be encouraged to do so.

The study was based on a concept, increasingly mentioned in the literature, that the “key” factor (Jeff Kramer’s term) [11] required for success in computer science is an aptitude for working with abstractions [3, 5, 10]. In one sense, this is ability, in a software design setting, to discern and create simplicity in the face of complexity. In another sense, it is an ability to conceptualize up the abstraction ladder to create software hierarchies of different varieties [5].

The intent of this investigation was to discover whether computer science students do, in fact, possess a stronger aptitude for abstraction than students in other majors. Researchers have investigated the aptitude for abstract thinking among computer science majors with exercises using concepts from computer science. No other researchers have attempted to gauge this ability across majors as is done here.

Because work up to now investigated the aptitude for abstract thinking among computer science majors with measures using concepts and terminology from computer science, it cannot be directly harnessed to measure the same aptitude among students who are pursuing other majors. This study has devised an instrument to measure abstraction aptitude among undergraduate students in all majors. The investigation attempts to determine
whether computer science students score higher on this measure than students in other disciplines.

2. CONCEPTUAL FRAMEWORK
This study was guided by recent research on abstraction as it is applied to computer science. Computer scientists, particularly those who focus on the teaching of computer science at the introductory level, have been working to develop an understanding of abstraction that would lead to an effective way of teaching students how to think abstractly [4, 5, 10]. Since abstraction is discussed in the literature as a thinking process, but also as an applied skill, how, then, does a student move from a conceptual understanding of abstraction to applying abstraction in the traditional Model, View, Control (MVC) levels utilized in object-oriented programming [11]? How can a researcher test and evaluate a student’s conceptual grounding in abstraction and ability to use abstraction in an applied way?

Orit Hazzan and Jeff Kramer are two researchers from the field of computer science who have written extensively on the need to develop an instrument to test abstraction [9, 10, 13]. Their work has examined the presence of abstraction in the computer science curricula as well as the challenges in trying to teach it at the undergraduate level. Their work has focused on the identification of the kinds of tasks that are particularly suited to abstraction and may therefore enhance students’ abstract thinking.

The conceptual framework for this study has roots in the study of abstraction as a cognitive ability, based upon the tests developed by Adey and Shayer in the 1980’s [1] in the United Kingdom and manifested in the recent work from Kramer and Hazzan [10, 13]. Their work has led to the identification of three scales of abstraction: conceptual abstraction, formal abstraction and descriptive abstraction.

Conceptual abstraction measures abstraction as it may be understood from an individual’s orientation within a big picture vs. small detail context. An individual with strong abstraction skills can move effectively between both orientations. Formal abstraction and descriptive abstraction flow from conceptual abstraction, as shown in figure 1, and are specializations in problem solving and language manipulation, respectively.

3. The Research Questions
The main focus of this study was to determine if computer science majors have stronger abstraction skills than other majors. The study will also test the model used to illustrate abstraction. The hypotheses are the following:

Hypothesis 1: Students who have declared computer science as their major will score higher than students in other majors on measurements of all three scales of abstraction.

Hypothesis 2: Conceptual abstraction (a self-reported inclination toward abstract thinking), is significantly related to formal abstraction (removing detail in order to simplify), operationalized as the ability to reason about symbolic situations or symbolic constructions in order to derive, deduce, or perceive underlying structure or a more straightforward (simplified) view.

Hypothesis 3: Conceptual abstraction is significantly related to descriptive abstraction (generalization so that one representation portrays multiple examples), operationalized as the ability to discern characteristics of chief importance and, therefore, to construct generalized accounts on models that are abbreviated and meaningful.

Hypothesis 4: The relationship between formal abstraction and descriptive abstraction is weaker than the relationship between either of these and conceptual abstraction. The rationale is that these are distinct manifestations of conceptual abstraction and the inclination toward either could account for a high score on the scale of conceptual abstraction.

4. RECENT RESEARCH IN ABSTRACTION AND COMPUTER SCIENCE EDUCATION
Noteworthy precursors to Kramer’s and Hazzan’s work in abstraction includes the study by Or-Bach and Lavy [14] and the study by Bennedsen and Caspersen [3]. Or-Bach and Lavy’s contribution is a typology for rating the quality of a derivation hierarchy designed by third year college students following the completion two semesters of coursework in object-oriented programming and design. The lowest quality design entailed an abstract class at the top of a Java derivation hierarchy that included only instance variables. In the middle quality design, instance variables and concrete methods had been factored into the top class. In the best design, the top class was an abstract class with instance variables, concrete methods, and abstract methods. Of the 33 participants, 4 constructed designs of top quality and 7 drew-up designs of middling quality. 20 of the
remaining 22 could only do a design of the lowest quality, and 2 could not even do this. Their recommendation was for instructors to increase the number of examples seen in class along with more drill and practice. For Or-Bach and Lavy, "abstraction" lies in the graduated factoring of classes up the design hierarchy, and they claim that the ability to accomplish this competently is teachable.

The Bennedsen and Caspersen study is interesting on several counts [3]. The first is the hypothesis, that abstraction ability is correlated with course grade in CS1. The second is the means used to measure abstraction ability, a test with eight gradations of "cognitive development" rooted in the developmental stages described by Jean Piaget. The third is the finding of no correlation between abstraction ability and course grade coupled to the explanations proffered to account for this.

Unlike Or-Bach and Lavy, for whom "abstraction ability" was literally tied to the design of an abstract class; for Bennedsen and Caspersen it pertained to a general mentality or thinking process, as in the interpretation of Knuth. Their study was the first empirical effort to explore the relationship of abstraction to students' success in CS1.

5. KRAMER'S MODEL AND QUESTION PATTERNS FOR MEASURING ABSTRACTION

This work shows that while Kramer's two-aspect model of abstraction is open to critique, it fits well with traditional conceptualizations, is particularly suited to the object oriented paradigm and aptly describes the application of abstraction in the field. Further, it is rooted firmly in Jean Piaget's description of cognitive development. Because Kramer's model appeared sound and eminently pertinent, it was adopted for this investigation.

Besides being theoretically acceptable, Kramer's model underlies the empirical work by Hazan and himself that yielded the "question patterns" (their term) for teaching and assessing abstraction. These helped to inform this research instrument and lend support to its validity.

Formal abstraction equates to "removing detail to simplify and focus attention." It is the term coined by the author for the scale comprised of problem-solving items requiring structure to be seen beneath irrelevancies. Descriptive abstraction equates to "generalization to identify the common core or essence." It is the term coined by the author for the scale comprised of items requiring interpretation and explication of most meaningful qualities.

As different as formal abstraction sounds from descriptive abstraction when they are defined, they are not mutually exclusive. Detail may be removed in order to generalize, or generalization may result from structural simplification.

5.1 Research Instrument Development

The main focus of this study was to determine if computer science majors have stronger abstraction skills than other majors. The goal was to develop a research instrument that could measure the relative abstraction skills of undergraduate students across disciplines to test hypothesis 1 from chapter 1: students who have declared computer science as their major will score higher than students in other majors on measurements of all three scales of abstraction.

There were challenges in developing such a test. The discussion in the computer science discipline about testing individuals for abstraction skill is an ongoing one and a standard for testing abstraction has not been agreed upon by the discipline. According to Kramer, "Unfortunately, we have been unable to find any existing appropriate tests. Tests for the formal operations stage focus mainly on logical reasoning and are not appropriate for testing abstraction skills nor distinguishing between the skills of students at college level [13 p. 41]." Research indicates that abstraction is not a single, monolithic concept, but rather a constellation of different, yet somewhat similar mental processes. In order to deal with this complexity, a means was sought to test the three elements of abstraction identified in this chapter: conceptual abstraction, formal abstraction and descriptive abstraction. By separating the overall concept of abstraction into these three cognitive areas, it becomes possible to develop measurements around the specific skill sets that are attributed to them. When the researcher says "testing abstraction," the reference is to measuring aptitude for constructing abstractions.

The survey developed and implemented for this study was designed to test abilities in the three scales of abstraction (conceptual, formal and descriptive) among undergraduate students.

5.2 The Setting

The University where the study took place is a comprehensive, independent institution in New York with campuses in New York City and Westchester County with approximately 13,463 students. The gender mix at Pace is 39 percent male and 61 percent female. The University is divided by discipline into six schools: The Dyson College of Arts and Sciences, The Lubin School of Business, The School of Education, The Lienhard School of Nursing, The Seidenberg School of Computer Science and Information Systems and the School of Law. Each school, with the exception of the School of Law, offers undergraduate programs. The institution awards associate, bachelors, masters, and doctoral degrees.

The survey for this study was made available to 560 undergraduate Pace University students in 32 class sections. These class sections can be formulated into three groupings: 1) a university core course in computing (CIS 101 - 210 students) taken by all students except computer science majors; 2) a computing course (CIS 102 - 292 students) with a prerequisite of CIS 101, that supports the university core but is not required; and 3) a required computing course (CS 121, CS 312, CS 389 - 48 students) or an upper level computing elective (CS 331 - 10 students) for computer science majors.

CIS 101, Introduction to Computing, the University core course in computing basics which is required of all Pace University undergraduate students (with the exception of computer science majors) focuses on building computing competency in three areas comprising data (Excel), Web development (HTML) and basic programming concepts (JavaScript). It is composed largely of freshmen and sophomores pursuing many different majors.

The CIS 102 courses included CIS 102Q - Problem Solving with LEGO Robotics, CIS 102 W - Web Design for Non-Profits, CIS 102T - Intergenerational Computing and one Honors course, CIS 396H - Problem Solving Using Technology and Human Endeavor in Urban Education. All of the CIS 102 courses focus on service
learning. In these service learning courses, students learn a series of computing-focused tasks at the beginning of the course and are then responsible for going out into the community to teach those skills to clients including school children or seniors, or to apply them in non-profit organizations. The student population ranged from sophomore through senior and students were drawn from many majors at the University, including computer science.

The third grouping is a selection of computer science courses from the undergraduate course cycle including CS 121 - Introduction to Programming, CS 312 - Research Methods in Computers and Society, CS 331 - Security in Computing and CS 389 - Software Engineering. The computer science courses were selected in order to provide a cross section of respondents who span the Pace University computer science majors.

A total of 227 surveys were returned from a total potential population of 560. Of those, nine were insufficiently complete for inclusion in the study and three were duplicates. This left a pool of 215 usable surveys returned from a population of 560 eligible students representing a return rate of 38.92 percent.

5.3 Theoretical Underpinnings of the Instrument

The instrument that was used was developed, in part, on the studies done by Kramer and Hazzan [9, 10, 13] and to some extent, Bernesden and Caspersen [12] and their work to develop an appropriate test of student abstraction skills.

In Hazzan and Kramer’s 2006 work [10], they suggest a series of question patterns as a means of testing abstraction in the computer science curriculum. They offer ten questions and recommend that an instructor adapt them to the context of the course that they are teaching. These questions were developed to present problem solving scenarios that require respondents to apply subject-specific abstraction skills in the manipulation and ordering of information. For example, the questions ask students to choose between two or more representations of a system and they ask them to choose the model that is more abstract, and then explain why it is more abstract. These questions are an excellent basis from which to develop questions to measure abstraction skills, particularly for the scale of descriptive abstraction. These questions were designed by the authors to be generic, adaptable and open-ended. However, they suggest caution in deploying these questions: “It is important to mention that such open questions require careful use so as to avoid a situation in which students give standard answers. We would not want our students, for example, to simply repeat considerations for using abstraction that they have heard in class [10, p.7].” These proposed questions were not ‘short answer’ and were not general in the sense of being usable for students in disciplines unrelated to computer science. As a result, these questions could not be used to test an interdisciplinary population of students for their potential as computer science students. In this light, it was incumbent to adapt these questions so that they would be suitable for a general audience (not just computer science majors) and that they be rendered in a format that is familiar to undergraduate students. In this context, it was deemed necessary to develop the survey questions in a multiple choice format and this, as with all of the survey questions for each of the three abstraction scales, proved challenging. A set of questions were developed to test and measure abstraction levels in a cross section of respondents from all academic majors where they had to analyze and prioritize information and identify those elements that were most important within the given context.

5.4 Developing Questions on Abstraction

The challenge, in lieu of an established and accepted means of testing abstraction, was to develop a means of testing abstraction skills among undergraduate students that would render feedback on their ability to abstract information in both a problem solving context as well as an information manipulation one.

The initial instrument was developed after a careful review of the abstraction literature. After review, 23 questions were identified for the final instrument. Prior to dissemination of the survey, an independent professional designer of psychological measurements and surveys endorsed the survey’s construct validity.

There are four sections on the survey including: background information, conceptual abstraction, formal abstraction and descriptive abstraction. The sections of the survey were organized in the following way: the background questions were designed to gather information regarding demographic characteristics of the student respondents. These included eight questions focusing, for each course taken, on which course they were enrolled in, current major, whether students had changed majors, if they had changed majors, what their previous major(s) had been, gender, age, year in school, and program at Pace (i.e. Honors, traditional, etc.).

The questions related to conceptual abstraction measure a self-reported inclination toward detail removal, simplification, and generalization. The three questions were designed to assess the respondents’ self-reported orientation on a continuum ranging from the specific, on one end, to the general, on the other. Students were asked to respond to a five-point Likert scale (strongly disagree, disagree, neither agree nor disagree, somewhat agree, strongly agree) to indicate a relative place on the scale.

One question was designed to measure the respondents’ interest in a variety of activities ranging from performing in public, to reading, to playing computer games. This question was included to gain some insight into a respondent’s orientation in terms of actions, activities and interests. Some of these interests and activities have been anecdotally associated with students in certain majors (i.e., political science students like to read periodicals).

Six questions were developed to test a student’s formal abstraction ability. These questions were problem solving questions that were oriented in logic and process. Several were slightly quantitative in nature. These questions challenged a student’s ability to work with more intricate, quantitative issues.

Three questions were developed that asked students to evaluate an information object in order to identify a respondent’s ability to focus on and feature the most important details in the object while ignoring or minimizing the less important ones. These questions, therefore, were designed to test the respondent’s descriptive abstraction abilities.

The survey instrument was developed, reviewed and then administered on paper to a section of CIS 102Q, Problem Solving with Lego Robotics. Twelve students took the survey and then were asked to discuss their reactions to the survey. Adjustments were made to the survey based on their feedback. The survey was then put into its initial electronic format and tested in a second service learning course, CIS 102W, Web Design for Non-Profits.
The students were, again, debriefed on their reactions to the survey instrument.

6. IMPACT OF DISCIPLINE ON ABSTRACTION SKILLS

Pace University has traditionally been a school for professional studies, and while the University is growing its enrollment in the arts and sciences, many of the respondents to the survey (more than 75%) were enrolled in Pace's pre-professional programs. In theory, each major at the undergraduate level develops a process of critical thinking that informs the problem solving skills and processes appropriate to that major. When majors are aggregated into disciplines, it is anticipated that the same or similar problem solving skills and processes are taught and shared. For example, a quantitative thought process is inculcated among the accounting and finance majors that is somewhat different from the quantitative skills taught to math majors or computer scientists. The students who have traditional quantitative course work have the opportunity to develop abilities that allow them to focus on problems of formal abstraction without experiencing the 'abstraction anxiety' that students lacking in-depth quantitative work often experience. Clearly students in every major develop skills that can be applied usefully in all three scales of abstraction. However, computer science majors seem to have an identifiable ability to function in all three scales of abstraction and to shift from a 'big picture' orientation to a 'small detail' orientation.

The literature strongly indicates that experience is important in the development of abstraction skills. 26.7% of the respondents were freshmen, students who had not yet been schooled in the abstraction skills relevant to their own discipline. This sparks the debate as to how much abstraction skill is inherent in the individual and how much it is a skill set learned by a combination of classroom work and experience.

6.1 Testing for the Three Scales of Abstraction

This study offered three scales of abstraction which research indicated may allow for a more specific and, perhaps, precise means of measuring and testing of abstraction skill: conceptual abstraction, formal abstraction and descriptive abstraction. The central focus was on measuring whether computer science majors have a higher level of abstraction ability than other majors.

Conceptual Abstraction The findings of this study indicate that conceptual abstraction is strongly related to the hypothesized abstraction scales: formal abstraction and descriptive abstraction. As shown in figure 1.1 in chapter I, conceptual abstraction is an umbrella skill that informs both formal and descriptive abstraction. Conceptual abstraction proved to be the only one of the three scales of abstraction that associates strongly with the two other scales, indicating that it serves as a foundation for abstraction ability. As in Kramer's observation [13, p.38] that his best computer science students have the best abstraction skills, it is indicated that conceptual abstraction associates strongly with discipline and with both formal and descriptive abstraction.

Formal Abstraction Formal abstraction equates to removing detail to simplify and focus attention. It is the term representing the scale comprised of problem-solving items requiring structure to be seen beneath irrelevances. The development of questions to test and measure formal abstraction abilities for this study was a challenging process and ultimately yielded the least definitive results. The attempt was a properly performed, first-ever try at building a brief, multiple-choice measure of an elusive cognitive ability. It was different in many ways from the instruments suggested by Kramer and used by others. The respondents indicated that the formal abstraction questions were challenging and therefore an acceptable representation of the types of questions and problem statements that can be used to test for formal abstraction. Formal abstraction associated strongly with conceptual abstraction. However, analysis showed that formal abstraction was weakly associated with discipline and with descriptive abstraction.

Descriptive Abstraction Descriptive abstraction equates to a process of generalization to identify the common core or essence. It is the term representing the scale comprised of items requiring interpretation and elaboration of most meaningful qualities. The research indicates that it is possible to measure an individual's ability to generalize or simplify in order to bring clarity to an item or object. Analysis shows that descriptive abstraction is strongly associated with discipline. It is also associated with conceptual abstraction. Discipline is not associated with formal abstraction.

The results of the study show a slightly different relationship between the three scales of abstraction. As shown in figure 4.1, below, while conceptual abstraction has been shown to be an 'umbrella scale' with which both formal and descriptive abstraction share association, formal and descriptive abstraction are unrelated except through the association with conceptual abstraction.

![Diagram showing the relationship between the three scales of abstraction](image-url)

Figure 2. The Relationship between the Three Scales of Abstraction Reflecting Hypotheses Testing

The results of the analysis of the survey responses indicate potential for measuring abstraction ability in individuals. The responses from our population allowed us to answer the hypotheses that were formulated in chapter I, though the outcomes were not always what was expected and were in some cases surprising.

Hypothesis 1: Students who have declared computer science as their major will score higher than students in other majors on measurements of all three scales of abstraction.

Finding: The performance of computer science majors on the survey instrument is the central focus of this study. Computer science students outperformed the other disciplines in answering questions designed to test each of the three scales of abstraction, but not as decidedly as the researcher would have predicted at the outset. The implications of this finding include support for Kramer's assertion about his best computer science students being
the most skilled at abstraction [13] and as well as a ratification of
the way that abstraction is taught to computer science students.

Hypothesis 2: Conceptual abstraction is significantly related to
formal abstraction

Findings: Results show that conceptual abstraction is related to
formal abstraction and serves as a foundation skill on which the
abilities of formal abstraction, related as they are to formal
problem solving, are able to thrive and function.

Hypothesis 3: Conceptual abstraction is significantly related to
descriptive abstraction.

Findings: As with its relationship to formal abstraction,
conceptual abstraction serves as the foundation skill set, or
gateway, in which descriptive abstraction skills can operate most
effectively.

Hypothesis 4: The relationship between formal abstraction and
descriptive abstraction is weaker than the relationship between
either of these and conceptual abstraction.

Findings: The research suggests that formal abstraction is only
loosely associated with descriptive abstraction. Indications are
that these two skills can operate efficiently and somewhat
independently without supporting skills from the other. However,
as noted in chapter II, as different as formal abstraction sounds
from descriptive abstraction, when they are defined, they are not
mutually exclusive. Detail may be removed in order to
generalize, or generalization may result from structural
simplification. The implications for this finding are that formal
and descriptive scales of abstraction are very different aspects of
the cognitive ability that we call abstraction. While both share
associations with the ‘umbrella scale’ of conceptual abstraction, it
is possible for students in different disciplines to be strong in one,
but not the other, while sharing a common grounding in
conceptual abstraction.

Discussion of the Hypotheses: The research confirmed the four
central hypotheses that are the foundation of this study.
Computer science majors do have stronger abstraction skills than
students in other majors. Conceptual abstraction is, indeed, a
foundation skill that is indicative of an individual’s abilities with
both formal abstraction and descriptive abstraction. Formal
abstraction and descriptive abstraction have shown to be abilities
that are exclusive of one another and that one may be strong in
one scale, but not in the other.

7. KEY FINDINGS OF THE STUDY

The study has shown that there is significance to the assertions of
Kramer, Hazzan, Caspersen & Bennedsen and others that
computer science majors have stronger abstraction skills and that
the best computer scientists may be the best abstractionists. In
addition, this study has shown the following:

- Although unconfirmed by final grades (because this was
  not methodologically feasible), Kramer’s assertion that
  abstraction may be the ability that separates the best
  students in computer science seems tenable. Discipline-
  specific abstraction skills may have a similar impact in
  other academic majors.

- Abstraction may be the ability that differentiates the
  mathematician from the computer scientist.

- If abstraction aptitude is key to success in computer
  science, there is absolutely no basis for presuming that
  one gender is better equipped for it than the other.

8. IMPLICATIONS OF THE STUDY

Leading computer scientists including Kramer, Hazzan, Denning,
Norman, Bergin, and others agree that the ability to develop
abstraction skill is integral to success in computer science and
software development. Researchers in the discipline of computer
science are striving to define abstraction as it pertains to computer
science while also working to develop a means for testing it and
developing it among students through course work. The results of
this study confirm that there is a real potential to construct
measures of abstraction that conform to accepted definitions and
may be applied to fulfill the educational utilities that Kramer and
Hazzan have seen for them.

It is also true that there is a troubling shortage of new talent
pursuing the field of computer science. Identifying individuals,
through testing, with strong abstraction ability is a tantalizing
prospect. Potentially successful computer scientists could be
identified among high school and college students.

Is there an ability that separates the skilled mathematician from
the skilled computer scientist? The results of this research seem
to indicate that, in fact, the level of abstraction skill may
differentiate the two. Many in the field have relied on
mathematics ability as an indicator of potential in computer
science. However, Caspersen and Bennedsen [3] and Ventura
[16] found that achievement in mathematics (such as SAT test
scores) was not an indicator of success in CS 1 courses. Knuth
[12] indicated that there was a difference between the skills that
are honed by mathematicians from those honed by computer
scientists.

These insights indicate that it is timely and necessary to develop a
means to test individuals on abstraction skills. Test sets could be
generic or discipline-specific and could be used to identify high
school students with potential in computer science, to test the
development of undergraduates as they move through their
courses, and to test the efficacy of mechanisms used in the
curriculum to teach abstraction.

9. CONCLUSIONS

Developing an abstraction curriculum and teaching
protocol that can be exported across disciplines has the potential
to be one of the computer science field’s great contributions to
academic discourse and scientific discovery. Abstraction is a
context dependant, yet widely accepted aspect of human cognition
that is vitally important for success in the study of computer
science, computer programming and software development. In
practice, abstraction makes the intangible and complex
understandable and malleable in ways that fuel the applied
development process. This study is a needed addition to the
literature, undertaken at a time when the potential for harnessing
the concept and application of abstraction have captured the
interest of many academics and practitioners in the field of
computer science. In undertaking this study, the researcher has
sought to develop a closer understanding of abstraction as it exists
in the literature, as it is applied in the discipline of computer
science and to test the efficacy of measuring it among a cross
section of undergraduate students. The three scales developed
within this study, conceptual abstraction, formal abstraction and
descriptive abstraction offer meaningful constructs in terms of
refining our understanding of the ways that abstraction can be
used as a problem solving tool.

The practical application of measuring abstraction skills
are significant and this study was both motivated and informed by
this potential. It revealed the feasibility of developing discipline-
specific abstraction tests. Such instruments can be valuable in
better understanding the abilities and needs of computer science
students and professionals as well as in identifying students with
the abstraction skills to be effective computer science
practitioners. Therefore, identifying abstraction ability in
individuals can be beneficial to the discipline of computer
science. The present work has shown that it is possible to
measure and compare abstraction skills between students in
computer science and students in other disciplines.

10. ACKNOWLEDGMENTS
My thanks go to Bernice Houle, Ph.D, Allen Stix, Ph.D and Dean
Susan M. Merritt, Ph.D, of the Seidenberg School of Computer
Science and Information Systems at Pace University for their
guidance and support in the course of this research.

11. REFERENCES

Cognitive Intervention and Academic Achievement,

(2006) "Social Networks Generate Interest in Computer
Science", Technical Symposium on Computer Science
Education, Proceedings of the 37th SIGCSE Technical
Symposium on Computer Science Education, 438-442.

Context – A Model-First Approach to CS1", Proceedings of
the thirty-fifth SIGCSE Technical Symposium on Computer
Science Education, 477-481.

Ability as an Indicator of Success for Learning Object-
Oriented Programming?", ACM SIGCSE Bulletin, vol. 38,
no. 2, 39-43.

Influence Success", Proceedings of the 36th SIGSCE
Technical Symposium on Computer Science Education, 411-
415.

Abstraction?", Proceedings of SIGCSE 2001, ACM, 26 – 30

Oriented Approach”, ACM SIGSOFT Software Engineering
Notes, March, vol. 28, no. 2

[8] Dijkstra, ("Algorithms in Modern Mathematics and
Computer Science" in Lecture Notes in Computer Science,
1981, volume 122, pages 82-99; also in the anthology
Selected Papers in Computer Science, Cambridge University
Press, 1996, Chapter 4, pages 87-114)

York State Education Summit, Albany, NY, 11/2/2005, from
IBM, p. 5

Learning Computation Theory Concepts", Annual Joint
Conference Integrating Technology into Computer Science
Education, Proceedings of the 7th annual conference on
Innovation and Technology in Computer Science Education,
156-160.

Science and Software Engineering: A Pedagogical
Perspective", System Design Frontier Journal, vol. 3, no. 12,
1-9.

Testing MVC and Workflow Based Web Applications,
Advanced International Conference on Telecommunications
and International Conference on Internet and Web
Applications and Services (AICT-ICIW06)

and Computer Science" Proceedings of the ("Algorithms in
Modern Mathematics and Computer Science" in Lecture
Notes in Computer Science, 1981, volume 122, pages 82-99;
also in the anthology Selected Papers in Computer Science,
Cambridge University Press, 1996, Chapter 4, pages 87-114)

Communications of the ACM, vol. 50, no. 4, 36-42.

Abstraction in Object Orientation: an Empirical Study, The

to CS1", Journal of the Consortium for Computing in Small
Colleges, vol. 17, no. 3 (February, 2002)

Identifying Predictors of Success for an Objects First CS1,
Diss. The State University of New York at Buffalo, (UMI:
Ann Arbor, MI).
The 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, Interim Dean, The Seidenberg School of Computer Science and Information Systems – Pace University
Susan M. Merritt, Computer Science, Pace—Westchester

Members:
Howard S. Blum, Computer Science, Pace—New York
Mary F. Courtney, Computer Science, Pace—Westchester
Nicholas J. De Lillo, Mathematics and Computer Science, Manhattan College
Daniel Farkas, Information Systems, Pace—Westchester
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. Mallozzi, 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 Seidenberg 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. 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 Seidenberg School of Computer Science and Information Systems
Goldstein Academic Center
Pace University
861 Bedford Road
Pleasantville, NY 10570-2799