DESIGN READINESS:
AN EXPLORATORY MODEL OF OBJECT-ORIENTED DESIGN PERFORMANCE

Tracy L. Lewis

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Applications

Manuel A. Pérez-Quiñones, Co-Chair
Mary Beth Rosson, Co-Chair
Joseph Chase, Committee Member
Helen Crawford, Committee Member
Stephen Edwards, Committee Member
Wanda J. Smith, Committee Member
David Tegarden, Committee Member

July 16, 2004
Blacksburg, Virginia

Keywords:
Academic performance, Cognitive abilities, Design readiness, Design, Individual differences, Object-oriented design, Prior computer science experience, Path analysis

UMI Number: 3241145

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

Design Readiness:
An Exploratory Study of Object-Oriented Design Performance

Tracy L. Lewis

ABSTRACT

The available literature supports the fact that *some* students experience difficulty learning object-oriented design (OOD) principles. Previously explored predictors of OOD learning difficulties include student characteristics (cognitive activities, self-efficacy), teaching methodologies (teacher centered, course complexity), and student experiences (prior programming experience). Yet, within an extensive body of literature devoted to OOD, two explanations of student difficulty remain largely unexplored: (1) varying conceptualizations of the underlying principles/strategies of OOD, and (2) preparedness or readiness to learn OOD.

This research also investigated the extent to which individual differences impacted DRAS and OOD performance. The individual difference measures of interest in this study included college grade point average, prior programming experience, cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, perceptual style), and design readiness. In addition, OOD performance was measured using two constructs: course grade (exams, labs, programs, overall), and a specially constructed design task.

Participants selected from the CS2 course from two southeastern state universities were used within this study, resulting in a sample size of 161 (School A, n = 76; School B, n = 85). School A is a mid-sized comprehensive university and School B is a large research-intensive university. If was found that the schools significantly differed on all measures of prior computer science experience and cognitive abilities.

Path analysis was conducted to determine which individual differences were related to design readiness and OOD performance.

In summary, this research identified that instructors can not ignore individual differences when teaching OOD. It was found that the cognitive ability visualization, prior OO experience, and overall college grade point average should be considered when teaching OOD. As it stands, without identifying specific teaching strategies used at the schools within this study, this research implies that OOD may require a certain level of practical computer experience before OOD is introduced into the curriculum. The cognitive ability visualization was found to have a significant indirect relationship with overall course grade through the mediating variable design readiness. Further, the results suggest that the DRAS may serve as a viable instrument in identifying successful OOD students as well as students that require supplemental OOD instruction.

DEDICATION

This dissertation is dedicated to all the individuals who prayed for me throughout my life. Although I have not always made the wisest decisions, it was the prayers of the righteous that allowed me to receive God's unmerited favor and complete this process.

AND

To my parents, Lee and Dyann Cotton, thank you for continuing to love me in spite of "me."

ACKNOWLEDGEMENTS

For the first four years, I ran the PhD race with my eyes on the prize of the "degree." In the last year I've learned that the degree is simply a by-product of the growth that one experiences, both intellectually and spiritually. I have to thank God for his presence in my life, as well as the many people that were sent to strengthen me along the way.

I would like to personally thank those individuals who supported me throughout my graduate school career. To my committee – Drs. Manuel A. Pérez-Quiñones, Mary Beth Rosson, Joseph Chase, Helen Crawford, Stephen Edwards, Wanda J. Smith, and David Tegarden – my sincere gratitude goes out to each of you.

My co-advisors – Dr. Rosson, thank you for your tutelage since 1999; and Dr. Pérez-Quiñones, thank you for picking up where Dr. Rosson left off.

Dr. Chase and Dr . Pérez-Quiñones, thank for the use of your class time and your students. You all made the commutes between universities almost worth it!

Drs. Crawford, Edwards, Smith, and Tegarden, thank you for your insight, feedback, and suggestions for improvements. Each added value to the dissertation and strengthened my skills as a researcher.

Dr. Wanda J. Smith, your direct way of telling me that "my stuff stinks" and to "just do it!" helped me pull through the most critical times of this dissertation process. The knowledge and wisdom that you possess is unfathomable.

A wholehearted *thank you* goes out to all of the students who endured the four or more hours of out-of-class time required to complete this study.

Thank you, Than Than Zin for your statistical support and inspirational moments.

Thank you, Laurie Good, for your hard work and dedication to editing.

Thank you, Robert Custer and his colleagues at Illinois State University, for the development and dissemination of the *Student Individualized Performance* rubric.

Now, to my friends, family: I needed to put this in writing because I know I don't say it often enough… I LOVE YOU!

Thank you, Harrington Family, for praying for me and praying with me when the days seemed the darkest. Vernard, thank you for being so supportive and attending my job talks and research defenses. You guys were there when I felt like no one understood the pain that I was going through. Thank you for assuring me that I didn't have to go through this process alone. Thank you for putting a mirror in my face to show me who I am and *whose* I am. Giovonni and Marquise, thank you for allowing me to regain and relive my youth!

Thank you, my best friend, my sorority sister, and my sister in Christ, Candi Cylar. You encouraged me to hang in there when I wanted to give up.

Thank you to all my friends, who put up with my five year stint of crankiness– Bruce Johnson, Yolanda Gibbs, Sandra McKinley, Rotunda Floyd, Erica Estep, Dr. Natascha Wilson, Dr. Darlene Eberhardt, Salem Baptist Church of Chicago, St. Paul A.M.E. Church, members of the Tau Mu Omega Chapter of Alpha Kappa Alpha Sorority, Inc., and the members of Roanoke Chapter of the Links, Inc.

Thank you to all those who came before me and paved that way for me to be where I am today!

And, last but not least, THANK YOU to my seasonal handyman, my brother and life-long friend, Larry Lewis.

Again, thank you to all those who prayed for this moment to arrive…

FINALLY, I BELIEVE!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

OVERVIEW

Although object-oriented software development has been evolving for over two decades, it has matured significantly over the past several years. Jia (2001) identified three influential factors that have contributed to the maturation of this technology: (1) the convergence of object-oriented modeling techniques and notations resulting in the *Unified Modeling Language* (UML) as the *de facto* standard, (2) the development of object-oriented frameworks and the widespread use of design patterns, and (3) the adoption of an object-oriented development paradigm by the software industry.

Riding on the wave of these advances, object-oriented technology—particularly involving design—has experienced unprecedented popularity. Not surprisingly, the rapid pace of development presents educational challenges to computer science and software engineering students (Jia, 2001). Indeed, the literature supports the fact that *some* students continue to have difficulty learning object-oriented design (OOD) principles (Buck & Stucki, 2000; Wallingford, 1996a). Previously explored predictors of OOD learning difficulties include student characteristics (cognitive activities, concept maps, self-efficacy); teaching methodologies (teacher centered, course complexity, overload), and student experiences (prior programming experience) (Tegarden & Sheetz, 2001; Buck & Stucki, 2000; Wallingford, 1998; Rosson & Carroll, 1997; Bergin, 1996; Maciel, Fernandez, and Garrido, 1996; Sheetz, Puhr, Nelson, and Monarchi, 1995; Linn & Clancy, 1992). Despite the extensive body of literature devoted to learning difficulties, two additional factors are still poorly understood: (1) varying conceptualizations of the underlying principles/strategies of OOD, and (2) preparedness or readiness to learn OOD.

Given the critical nature of OOD principles and their importance for use in software development (Northrop, 1993), students must be able to fully comprehend object-oriented design concepts, as well as be able to efficiently apply them. However, an extensive literature search and personal observations have shown that some undergraduate students continue to experience difficulty understanding and retaining the design knowledge and skills taught during classroom activities. (Ventura, 2004; Astrachan, 2001; Bagert, 1996; Clancy, 1996; Maciel, Fernandez & Garrido, 1996; Rappin, 1998; Wallingford, 1996; Booch, 1994; Wirfs-Brock, Wilkerson & Wiener, 1990).

Research Rationale

The IEEE/ACM Joint Task Force on Computing Curricula 2001-Computer Science (CC2001) introduced the objects-first approach that "emphasizes the principles of object-oriented programming and design from the very beginning. … The first course … begins immediately with the notions of objects and inheritance … the course then goes on to introduce more traditional control structures, but always in the context of an overarching focus on object-oriented design" (Ventura, 2004, p. 1). Although CC2001 provided explicit instructions to include OOD in course curricula, there was no specific listing of OOD topics that should be covered in the course.

Today's undergraduate OOD educators are faced with two difficult challenges: (1) determining *what* concepts of OOD should be taught, and (2) understanding *why* some students struggle to learn OOD. The present research is an attempt to address both of these challenges.

Problem Statement

The premise that some students are more likely to succeed than others in the area of procedural programming has been under examination for decades. Numerous studies have reported significant linkages between programming success and a variety of demographic, attitudinal, and personality variables. For example, some scholars have maintained that gender is a moderator variable of course performance (Margolis & Fisher, 2002; Pearl, Pollack, Riskin, Thomas, Wilson, 2000; Liu & Blanc, 1996). Other researchers have found that motivation positively affects performance in a student's first computer science course (Reif & Kruck, 2001;Wilson, 2001). In fact, intrinsic motivation has been suggested to be a major factor contributing to the relationship between demographic variables and course performance. As another example, cognitive abilities have been linked to programming success (Hahn, Hahn, & Kim, 1997; Ryan & Al-Qaimari, 1996; Witkin, Moore, Goodenough, & Cox, 1977). Specifically, these studies have shown that logical reasoning is linked to success in procedural programming.

In contrast to the vast body of literature on programming success, only a few studies have examined factors associated with OOD performance. One such report investigated predictors of success in an object-first programming course (Ventura, 2004). Ventura explored the linkages between prior experience and course performance and concluded that prior programming experience was not a consistent predictor of object-first programming success. Instead, Ventura indicated that a student's effort and comfort level were more reliable predictors of success. While Ventura was one of the first researchers to examine aspects related to object-first programming course success, there is no research on identifiers of OOD success.

Moreover, there is a need to identify when a student is *ready* to learn design. This research has resulted in the concept of design readiness to refer to a student's potential to understand design principles. More specifically, it is the initial point and transitional process of coupling a student's prior experience with design specific training.

In conjunction with design readiness, there is also a need to identify the background characteristics associated with a "good" OOD student. If the characteristics of a potentially successful OOD student can be identified, instructors can then aid students in designing an appropriate program of study, and, ultimately, identifying a suitable career path. Additionally, identifying characteristics of OOD students can potentially assist instructors with identifying those that are not design *ready* and offer supplemental instruction to achieve the *maturity* needed OOD.

Purpose of the Study

The purpose of this research is to:

1. Develop an OOD problem-solving model, outlining the progressive stages and principles needed for teaching and learning OOD.
2. Construct and validate an instrument designed to measure object-oriented design readiness, namely, the Design Readiness Assessment Scale (DRAS).
3. Identify experiences and cognitive measures that are associated with object-oriented design performance. Experience measures of interest include prior computer science experience and college grade point average. Cognitive measures will include tests of spatial ability, visualization, perception, logical reasoning, and flexibility of use.

Research Questions

This research evaluates the concept of object-oriented design readiness, and will address the following five questions:

1. What student characteristics (i.e. background and/or cognitive state) are related to pre-training OOD abilities?
2. What individual differences are related to performance in an OOD course?

3. What individual differences are related to performance on a design task?

4. Is pre-training design ability related to a student's performance in an OOD course?

5. Is pre-training design ability related to a student's performance on a design task?

Figure 1 depicts the relationships among constructs of interest in this study. Within this research individual differences will refer to measures of demographics, academic achievement, and cognitive abilities. Pre-training design abilities will refer to measures of readiness and the pre-training design task. OOD performance will refer to the overall OOD course grade and performance on the post-instruction design task.

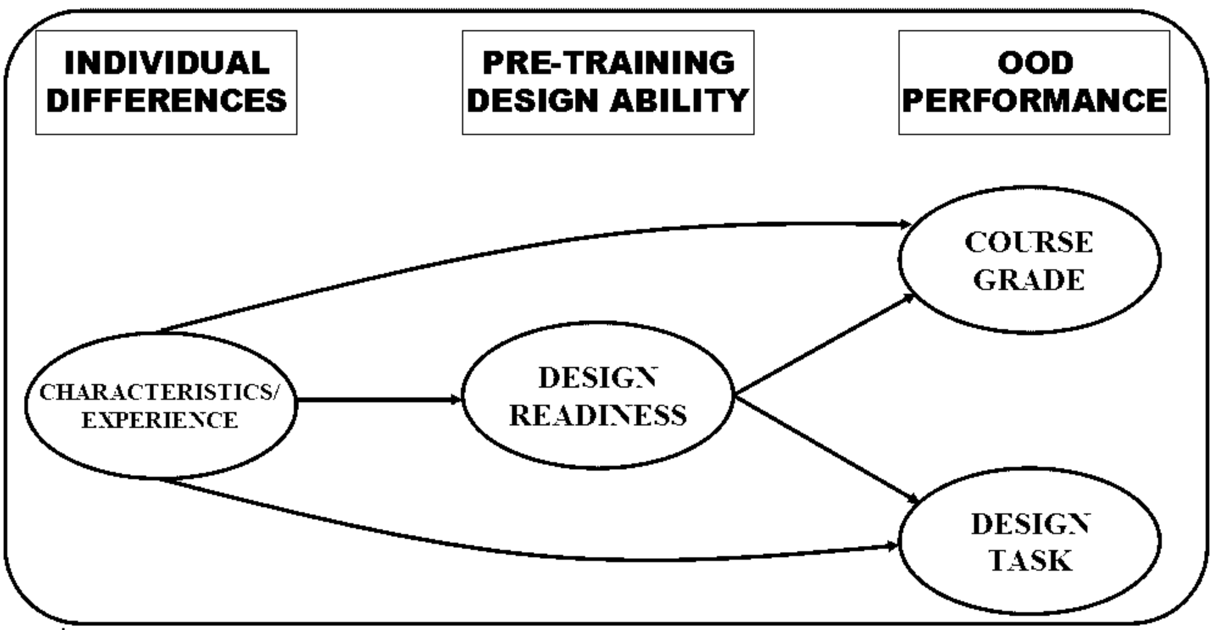


Figure 1.

Model of Research Questions and Contributions

The current research will attempt to identify significant relationships between individual differences and OOD performance. The concept of design readiness will serve as the moderator in the overall success of a student in an OOD course. While existing literature supports linkages between individual differences and course performance, no studies exist that examines the relationship between design readiness and design performance. Thus, this research attempts to examine the relationships among individual differences, design readiness, and OOD performance.

Terminology

It is important that key terms used in this study are clearly defined. Although the terms listed below are commonly used in object-oriented design research, it is necessary to explain each term in the context of this particular study.

Object-Oriented Design. (Bridle, 2000): An approach to software design in which a system is modeled as a collection of co-operating objects. This term also refers to the analysis of a problem area to identify its representation in terms of objects, and the specification of the ways in which they must co-operate.

Object-Oriented Programming. (W3C, 2004): A programming technique that speeds the development of programs and makes them easier to maintain through the re-use of "objects" that have behaviors, characteristics, and relationships associated with them; the objects are organized into collections, which are then available for building and maintaining applications

CS2 Course (ACM/IEEE Computing Curricula, 2001): Traditionally CS2 refers to the second course within the computer science curriculum; however within this research, CS2 will refer to the second course within the ACM/IEEE *objects-firsts* computer science curriculum. The course continues to introduce more complex control/data structures in the context of an overarching focus on object-oriented design.

Cognitive Ability. (Gall, Borg, & Gall, 1996): The process or ability of knowing by thinking, comprehending, analyzing, or evaluation. Cognitive ability relies on the process of gradually building one's understanding of the world through experience and maturation.

Prior Computer Science Experience. Prior computer science experience is characterized as the self-reported level of experience to various programming languages, operating systems, development applications, and general computing resources.

Significance

This research introduces an OOD problem-solving model that has been adapted from Polya's (1957) mathematical problem-solving model. The OOD problem-solving model is then actualized through the creation of the Design Readiness Assessment Scale (DRAS). There are two attributes of the OOD problem-solving model that will be helpful to instructors. First, it explicates what design skills a student needs to be able to solve a particular stage of the design problem. As noted earlier, the term *skills* refers to design strategies/principles. Second, it is purposefully language-independent, allowing instructors to have freedom in design problem development.

This research has the potential to make significant contributions to the literature on factors associated with OOD performance. First and foremost, there is a lack of reliable information focusing on when a student is *ready* to learn OOD. Thus, this research is the first known attempt to develop a model of object-oriented design performance moderated through what will be referred to as *design readiness*. The current research is interested in identifying pre-training characteristics (college grade point average, prior computer science experience, cognitive ability, and/or design readiness) associated with OOD course performance, as well as completing a specially constructed design task.

The findings of this study may aid school personnel in (1) identifying students most likely to succeed in OOD and (2) identifying those students who may require supplemental instruction in OOD; assisting students and academic advisors in the selection academic courses that will further enhance their likelihood to succeed at the highly prized industrial skill, OOD.

Summary

The chapter identified two unexplored difficulties of learning OOD: (1) varying conceptualizations of the underlying principles/strategies of OOD, and (2) preparedness or readiness to learn OOD. The problem statement addressed the importance of expanding the available OOD literature to include factors associated with design readiness and OOD performance, which helped conceptualize the purpose of the study. The quantifiable research questions were stated and terminology clearly defined. This chapter concluded with statements of potential significance of this research.

The remainder of this dissertation is outlined as follows. Chapter 2 presents a model of OOD problem-solving, which was used to develop the Design Readiness Assessment Scale (DRAS). Chapter 2 then discusses prior research on the conceptualization of OOD and predictors of performance in computer programming. Chapter 3 presents a research study that uses the DRAS and various measures of individual differences in an effort to identify measures associated with OOD performance. Chapter 4 presents the results of analyses of the data using multiple regression and path analysis. And the dissertation concludes with Chapter 5 presenting the interpretation of the results, limitations, and future work.

CHAPTER TWO

LITERATURE REVIEW

Given the critical nature of OOD principles and their importance for software development (Northrop, 1993), students must be able to understand, appreciate, and apply these concepts in meaningful ways. However, personal observations and a comprehensive literature search (including related Ph.D. dissertations) revealed that some undergraduate students continue to have difficulty understanding and retaining the design knowledge and skills taught during classroom activities (Ventura, 2004; Astrachan, 2001; Bagert, 1996; Clancy, 1996; Maciel, Fernandez, & Garrido, 1996; Rappin, 1998; Wallingford, 1996; Booch, 1994; Wirfs-Brock, Wilkerson, and Wiener, 1990).

This research will review the characteristics of OOD, as well as identify the attributes of students most likely to be successful. Furthermore, this study seeks to confirm the need to develop a sound OOD problem-solving model that will promote successful teaching and learning of OOD principles/strategies.

Given the paucity of literature related to predicting OOD readiness and course performance, this chapter will review related performance prediction literature. This associated literature includes studies predicting performance in object-first programming, engineering design, and in the first course of an information processing course of study.

Object-first programming prediction literature is presented as supporting literature because it is the *natural* counterpart to OOD. The IEEE/ACM Joint Task Force on Computing Curricula (CC2001) suggested that in early courses, objects-first programming and object-oriented design should be taught simultaneously. This was recommended as a technique to convey the importance of coupling OOD concepts and actual implementation.

The available literature relevant to first course performance in an information-processing plan of study was also examined. Specific areas of interest included software engineering, information systems, and computer science. In many universities these courses are prerequisites for any introductory OOD course.

In addition, this research assessed the literature relevant to introductory engineering design courses. This was deemed appropriate because engineering design follows a regimented procedure similar to OOD. Specifically, the process of design in both areas includes identifying the problem, breaking it into manageable chunks, rationalizing between decision choices, and selecting an optimal solution.

The remainder of this chapter will define OOD, characterize OOD readiness, present the theoretical framework for an OOD problem-solving model, and identify studies that may provide insight into identifying characteristics associated with OOD readiness, course performance, and design task performance. This review will specifically examine gender, prior computer science experience, academic performance (college grade point average), and cognitive abilities.

Defining Object-Oriented Design Principles/Strategies

Pioneers of OOD envisioned it as a language-independent, straightforward process of decomposing complex system requirements into manageable components. Booch (1994) suggested that the OOD process begin by simply searching for objects, which was achieved by identifying the "nouns" in a problem scenario. He further stated that one could identify the semantics of the objects by reviewing the corresponding verbs and the interactions with the other nouns (objects). Booch maintained that a quality OOD was achievable if one began with the mindset of developing highly cohesive and loosely coupled systems.

Others, however, were not convinced of the simplicity of OOD. Northrop (1993) called the design a “black-art” and argued that many thought the very idea of a design approach was an anathema. Northrop elaborated that producing a design that captured the various requirements and constraints was a complex task, only mastered through practical experience. Rosson and Carroll (1997) agreed with Northrop and stated that OOD was a complex process, requiring several spiral iterations to complete.

This research study takes a more moderate stance on the simplicity/complexity of OOD. It purports that OOD is an understandable activity if one begins with the fundamentals. The term “fundamentals” refers to those principles that (1) provide a simple cognitive mapping from previously learned programming techniques, (2) allows one to model system complexity, (3) facilitates in managing tradeoffs between design decisions, and most importantly, (4) provides language dependent representations of the system. Jia (2003) and Wirfs-Brock (2003) have identified a fundamental set of principles/strategies that can be applied to any design problem to yield a robust solution.

A brief summary of these principles/strategies is provided below. It should be noted that there will be a noticeable overlap in the definition of these principles as well as their applicability in object-oriented design and coding. From a constructivist point of view, this overlap provides the transitional support needed to mentally bridge the principles to form a foundation for teaching and learning OOD.

Divide and Conquer

The first step in designing a successful program is to divide the overall problem into a number of modules that will interact with each other to overcome the problem. In short—identify smaller components of the problem and solve them separately. Employ a division of labor much as we do in organizing many of our real world tasks.

Encapsulation

Once the modules or classes are identified, the next step involves deciding for each class what attributes it has and what action it will take. The goal is to place within each class the appropriate combination of data and functionality into a single entity, with the implementation hidden from external entities. Each class is designed to be a self-contained module with a clear responsibility and the skills (attributes and actions) necessary to carry out its role. In addition to knowing how to perform its role, each class has to know exactly what information it needs to obtain from its collaborators (internal and external system classes).

Generality

As long as we are designing a class to solve a particular problem, we should design it to be as general as possible. To design a general class requires a great deal more thought and effort than designing a narrow, single purpose class. Although it may not initially be obvious how specific problems might stimulate general solutions, we should design classes not for a particular task, but rather for a particular *kind* of task.

Information Hiding

The details of each class's performance should be hidden from other classes. This strategy will help classes work together cooperatively and efficiently. It should be noted that this technique is different from encapsulation because encapsulation involves bundling data with the precise methods that operate on the data. Conversely, information hiding involves hiding difficult design decisions or design decisions that are likely to change. We should hide information in a manner that isolates clients from requiring intimate knowledge of the design, and from the effects of changing design decisions.

Inheritance and Polymorphism

In general terms, inheritance and polymorphism manages a class's ability to use properties and abilities of another class (parent or super class), while adding its own functionality. More specifically, polymorphism may be understood in terms of inheritance as a means to define the properties of a subclass, and by adding information delimiting a subset of the elements corresponding to the parent class.

Interface

In order for classes to work cooperatively and efficiently, we have to clarify exactly how they should interact or interface with one another. An interface is a contract in the form of a collection of methods and constant declarations. When a class implements an interface, it promises to implement *all* of the methods declared in that interface. A class's interface should be designed to protect its integrity and to constrain the way the components of the class can be used by other classes.

Abstraction

Taken individually, each of the preceding principles provides a manifestation of the more general principle of abstraction. An abstraction denotes the essential characteristics of a class that distinguishes it from all other kinds of classes and thus provides its essential behavior, and nothing more. Abstraction is the ability to group large quantities of information into a single chunk. Organizing a complex set of attributes and actions into a single class and then dealing with the module as a whole is a form of abstraction.

Given these basic principles, this researcher defines object-oriented design as the decomposition of a problem area to identify its representation in terms of objects/classes, and the specification of the ways in which they must co-operate. This definition will facilitate *what* and *how* OOD is taught and learned.

Defining Readiness

Two well-validated forms of readiness are represented in the literature: *mathematical* (Heinze, Gregory, Rivera, 2003; The Berkley Math Readiness Project, 2003) and *reading* (Harvard University, 2003; Matthews, Klaassens, Walter, & Stewart, 1999). The concept of readiness has also sparked interest in other areas, such as science (Orsak & Acosta, 2002), the Internet (Arsin Corporation, 2001), and design (Lewis, Pérez-Quiñones, and Rosson, 2004).

Reading Readiness

Simply stated, reading readiness is when a child is prepared to profit from beginning reading instruction, typically measured by his/her ability to decompose words into constituent sounds (Matthews, Klaassens, Walters, & Stewart, 1999). More specifically, reading readiness is "a transition extending over several months during which time the child (student) gradually changes from a non-reader to a beginning reader. In this case, the readiness program couples the (student's) past learning with new learning and brings the (student), gradually, through the transition" (Harvard University, 2003, p. 1), The concept of design readiness was created by analogy—an initial point and transitional process of coupling the student's past learning with new learning, gradually increasing his/her level of design knowledge.

Mathematical Readiness

Early childhood mathematical readiness refers to a child's preparedness to move from the fundamentals of mathematics (counting, number order, shapes, addition, and subtraction) to more complex manipulations (fractions, multiplication, and division) (Onish & Del Sur, 2000; The Berkley Math Readiness Project, 2003). There are also pre-college initiatives (University of Vermont; University of Arizona) that require a mathematical readiness test to assess a student's mastery of pre-college mathematics fundamentals.

While the above referenced literature was consulted in the development of an operational definition for design readiness, literature sources that were more pertinent to math readiness and implications for success in engineering and computer science (Mayer, Dyck, & Vilberg, 1986; Heinze, Gregory, & Rivera, 2003) were also examined. While these studies did not examine the antecedents to math readiness, it was found that math readiness was associated with overall academic performance.

Design Readiness

This study is the first known attempt to identify antecedents of readiness – particularly design readiness. Lewis, Pérez-Quiñones, and Rosson, (2004) offer an initial definition of the concept, suggesting that design readiness can be viewed as a detailed snapshot of one's design potential. More specifically, design readiness is the point at which one is able to understand the concepts of OOD. Design readiness couples one's past experiences with new instruction and gradually transitions her/him to a higher level of design ability. A student's ability to recognize and apply OOD principles is both the beginning phase and transitional segue into design readiness. The following section will present an OOD problem-solving model that can potentially guide students in their learning process, as well as guide instructors in their dissemination of OOD instruction.

The Development of an OOD Problem-Solving Model

Over the years, many scientists and mathematicians have studied the phenomenon of problem-solving ability. A common thread throughout these studies is that successful problem-solving involves a series of stages. These stages reveal the thought processes, actions, and events that take place in order for an individual to devise a solution to a given problem. Polya (1957) developed one of the most widely accepted problem-solving models (1957). Polya's problem-solving model can intuitively be adapted to include the seven principle of OOD. This research will briefly describe the key elements of this model before adapting it to the OOD arena.

Polya's Problem-Solving Model

Polya's (1957) classic mathematical model listed four stages of problem-solving: (1) understanding the problem, (2) devising a plan, (3) carrying out the plan, and (4) looking back. The following subsections provide a more detailed description of Polya's problem-solving model.

Understanding the Problem. Stage One of the problem-solving model advised students to understand the problem. Students should be able to answer questions such as (1) What is the unknown? (2) What are the data? (3) What is the condition? (4) Is it possible to satisfy the condition? and (5) What are the limiting circumstances that one must work around? Additionally, in this first stage Polya advised that any given problem be separated into identifiable parts. Additionally, Polya recommended that the student draw a figure or picture, or introduce some kind of notation to visualize the question.

Devising a Plan. Stage Two of Polya's model involved identifying the connection between the data and the unknown. To facilitate this process, Polya recommended that students think of a related problem either in the same form or a slightly different form. Moreover, he suggested that students ask three questions in this phase: (1) Can you imagine a more accessible, analogous, general or special problem? (2) Can you envision a subset of the problem? and (3) Have you taken into account all the essential components involved in the problem? Polya stated that through the process of identifying related problems, the student might be able to reuse methods or parts of a previously successful plan. Next, Polya suggested that the student decide on the calculations, computations, or constructions needed to devise the plan. Lastly, in this stage students are advised to make sure that all appropriate data and conditions are considered.

Carrying Out the Plan. Stage Three of the problem-solving model required the student to perform all the necessary calculations and to check them as they proceeded. Polya proposed three questions to facilitate the execution of the plan: (1) Can you see clearly that the solution is correct? (2) Can you prove that it is correct? and (3) Can you combine calculations to represent a simpler solution?

Looking Back. The final stage of the Polya's problem-solving model recommended that students take time to reflect on their understanding of the solution. Students were also advised to reevaluate the plan that was carried out. Recommended questions students should ask are: (1) Can I get the result in a different way? And (2) Can I use this approach to solve another problem? Polya believed that following these stages would enable students to identify useful strategies when solving future problems.

Given the simple and intuitive nature of Polya's four stages, this paper suggests it may be an excellent framework for teaching and learning OOD principles. How it can be modified for the OOD context is described below.

Proposed OOD Problem-Solving Model

Figure 2 depicts an adaptation of Polya's model. This adapted framework includes the seven previously discussed OOD principles/strategies. How each problem-solving stage is related OOD principles/strategies will be discussed.

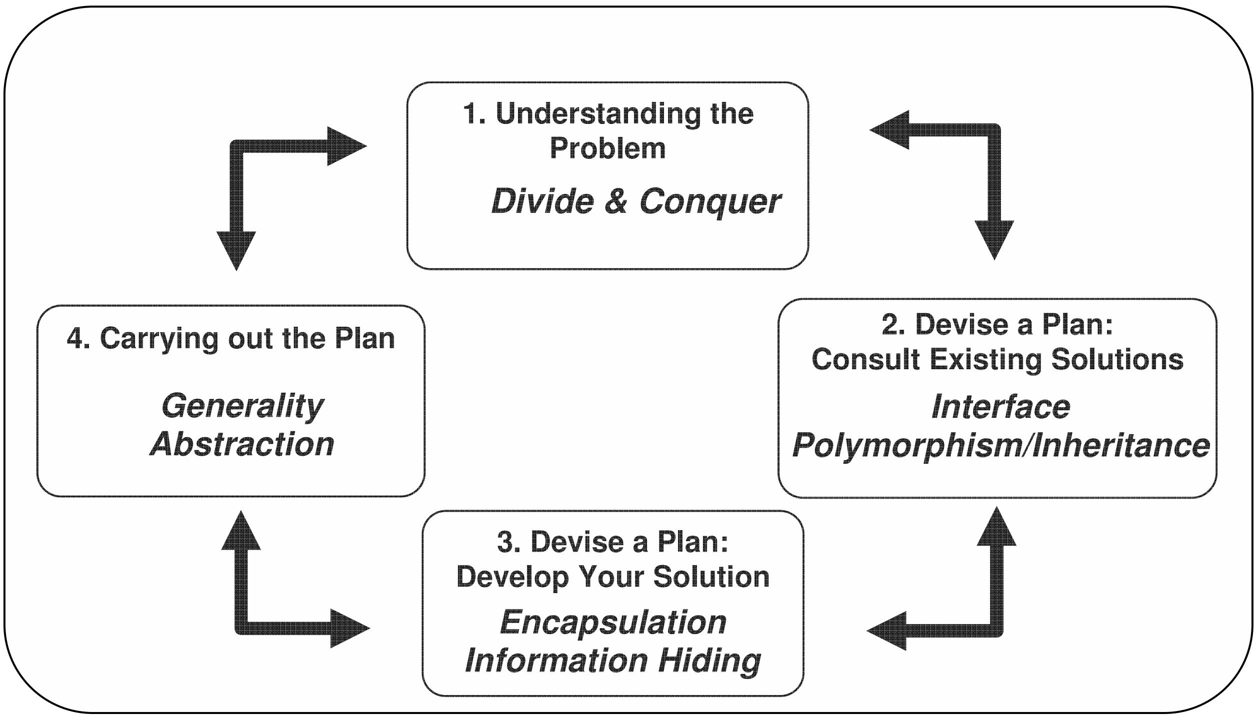


Figure 2.
Proposed OOD Problem-solving Model

<u>*Stage One: Understanding the Problem*</u>. Polya's problem-solving model suggested that the student understand the problem by partitioning it into identifiable chunks, which can further be divided into system components or classes based on functionality. This is analogous of the OOD principle of *Divide and Conquer* in that employing a division of labor process can simplify the problem.

Stage Two: Devise a Plan – Consult Existing Solutions. In this devise-a-plan phase, students consult existing and proven solutions to assist in developing a resolution to the new problem. When applied to OOD, consulting existing solutions may require the use of existing libraries, classes, or templates (*Interface and Polymorphism/Inheritance*). Referencing existing system libraries or templates aids in the development classes, abstract data types, and simply displaying text. The use of an existing interface provides the outside views of the class while hiding its internal behaviors. The interface primarily consists of the declaration of all the operations applicable to instances of the class (Booch, 1994). The use of polymorphism/inheritance provides freedom to share behaviors across classes, as well as denotes a single name to refer to this behavior. In this stage, classes are developed without regard to every detail of the data structures. This simplifies the overall design because one is not focused on data manipulation details that may change.

Stage Three: Devise a Plan – Develop Your Own Solution. This stage corresponds to the second phase of Polya's devise-a-plan, namely, decide on the calculations, computations, or data. In OOD, developing a solution entails protecting and placing the combination of data and functionality into a single entity (i.e., *Encapsulation and Information Hiding*). Stage Two (*Interface and Polymorphism/Inheritance*) dealt with the externals of the class structure, whereas this stage deals with the internal data manipulation. The internal data structures are created and shielded from external violation. This is a necessary task in OOD as system complexity is constantly changing and there needs to be means in places to alter data without disrupting the entire system.

Stage Four: Carrying Out the Plan. Carrying out the plan phase involves developing a clear and concise solution through combining procedures that will simplify the resolution (*Abstraction, Generality*).

The final stage of Polya's problem-solving model is applicable throughout the entire design process since designers are constantly required to reflect on design tradeoffs and proposed solutions. Therefore, bi-directional arrows connecting the four phases of the OOD problem-solving model represents the looking-back phase (Figure 2).

Linn and Clancy (1992, p. 122) stated that, "Instructors often assume that students can take their general problem-solving skills and discover specific software design skills on their own. Thus students learn design skills through unguided discovery." There seems to be an assumption that *if* students learn to program by writing codes of increasing size and complexity, they will *then* implicitly discover and apply the necessary design strategies. However, students are failing to demonstrate an understanding of the connection between object-oriented programming structures (e.g. objects, classes, methods) and the higher-level design strategies (e.g. abstraction, composition, design patterns) (Buck & Stucki, 2000; Kafura, 1998; Astrachan, 1996). The OOD problem-solving model created in this research can potentially guide students and instructors in OOD pedagogy.

There are two attributes of the OOD problem-solving model that are helpful to instructors. First, it explicates what design *skills* a student will need to be able to solve a particular stage of a design problem. The term "skills" refers to the previously described design strategies/principles. Second, it is purposefully language-independent, allowing instructors to have freedom in design problem development.

The OOD problem-solving model was used to guide the development of a measure of OOD readiness—the Design Readiness Assessment Scale (DRAS).

Design Readiness Assessment Scale (DRAS) Development

The DRAS was developed to measure one's pre-training design ability in an effort to assess its association with design task performance and performance in an introductory OOD course. Each question required the application of one of the previously mentioned OOD principles/strategies.

In order to develop concept-specific questions without the constraints of a specific programming language, the DRAS questions were developed using real-world problem scenarios. The use of real-world problem scenarios is employed by companies such as Microsoft® in their Certified Solution Developer (MCSD) and Academic Learning Series (ALS) to accurately test a candidate's ability to analyze and synthesize information and generate solutions (Microsoft, 1999). The use of real-world scenarios is also a common testing format utilized for many of today's standardized tests, such at the ACT and SAT college entrance exam, as well as the GRE graduate school entrance exam.

Test Construction Process

The DRAS was envisioned to serve as a general measure of OOD learning aptitude in universities offering courses in OOD. Because of this, it was essential to follow widely accepted testing construction strategies used in the development of standardized tests. Table 1 describes the steps involved in creating a standardized test using real-world problem scenarios. The information was interpreted from the works of Aiken (1998) and Svinicki (2002).

Table 1.

Steps Involved in Creating a Standardized Test Using Real-World Problem Scenarios[1]

STEP	PROCESSES
1	Outline Test Objectives
	Identify any subscales
	Translate concepts into actual measures
2	Create a Blueprint
	Create a detailed table of specifications against some level of cognitive difficulty or depth of processing
3	Item Preparation
	Decide on question format
	Operationalize the blueprint for each subscale
	Create plausible distracters
4	Directions
	Create unambiguous directions
	Explain the purpose of the test (verbal or written)
	State estimated completion time
	State how the responses will be scored (is guessing allowed?)
5	Test Evaluation
	Administer the test to subject matter experts or a focus group
	Evaluate for item difficulty
	Identify possible confusing questions
	Identify missing concepts
	Rewrite test questions (if needed)
	Re-evaluate test with focus group (if needed)
6	Validity Generalization
	Identify what is needed to administer, score, and interpret the results of the test in other situations
	Evaluate test results in multiple situations

[1]*Note:* Interpreted from *Tests and Examinations: Measuring Abilities and Performance* by Lewis Aiken *and Test Construction: Some Practical Ideas* by Marilla Svinicki.

Step 1. The test creation process begins by identifying the test objectives in a straightforward manner. The fundamental concepts of OOD (*Divide and Conquer, Encapsulation, Interface, Information Hiding, Generality, Inheritance and Polymorphism*, and *Abstraction*) will serve as the measures represented by subscales within the DRAS. Because the DRAS contains a real-world scenario for each of the questions, it was necessary to limit the number of questions to control for subject fatigue. The end result was a 28-question scale; allowing the DRAS to be divided into seven subscales of four questions each.

Step 2. In order to create a blueprint for DRAS item construction, the OOD problem-solving model was consulted. Each DRAS was created with no assumption of prior experience. The DRAS questions reflected a specific OOD principle/strategy as depicted in the OOD problem-solving modeled. The ultimate goal of the DRAS was to assess an individual's ability to break down the problem scenarios (*Divide and Conquer)* into understandable system components (*Inheritance and Polymorphism*, *Interface*) that would interact with each other without compromising system integrity (*Encapsulation, Information Hiding)*, ultimately creating a highly cohesive, loosely coupled system (*Abstraction, Generality).*

Step 3. During the item preparation stage, real-world scenarios were used to capture the essence of complex design problems. This was an enormous undertaking. It first required the development of a language dictionary, described below, of common words and phrases that would be roughly analogous to the technical OOD terminology. A team of local experts, consisting of graduate students and faculty, were consulted to evaluate the "leap" between OOD principles and common terminology. Participants were asked to evaluate the dictionary in terms of completeness, consistency, and unmerited leaps between principles and terms.

The language dictionary featured a list of common words and phrases for each of the OOD principles. Figure 3 provides a snapshot of the language dictionary used to create the DRAS, as well as how the key words were mapped into an *OOD* real-world scenario and questionnaire. The example shown represents the *divide and conquer* strategy. The phrase "things to do" was taken from the dictionary of terminology and was used to develop this scenario. The principle of divide and conquer requires one to (1) be able to recognize that the system should be divided into components, (2) be able to identify smaller components of the problem, and (3) be able to solve them separately. Divide and conquer employs a division of labor similar to what an individual may confront in organizing any number of real-world tasks. The example included in Figure 3 requires that the examinee identify all the necessary task that Jenny was required to complete and then decide on a strategy as to how she should complete the tasks.

Divide and Conquer	**Encapsulation**	**Generality**	**Information Hiding**
List of Tasks Order Group Things to do Perform tasks Assign tasks	Instruct Issue Inform Duties Roles Chores	Template General Generic Customize Arbitrary	Hide Contact Handle Present Give Request
Inheritance and Polymorphism	**Interface**	**Abstraction**	
One operation / Multiple tasks Same name, different meaning Specialized Specific Take information, pass information on Modify existing	Plan Front end Build No interruption in flow of control Seamless transition	Organizing Functionality Features Difference	

Jenny has a million things to do before she catches her plane tomorrow morning. She is remembering that she has to wash her hair, iron her clothes, take the dog to the kennel, write checks for her bills, check the weather, check the status of her flight, call her mom, paint her nails, pack, pick up the rest of her clothes from cleaners, and take out the trash.

Q. How should Jenny go about doing all of the things she has to do?

a. Do things as she remembers them.
b. Start with the task that requires the least amount of energy or time and work towards the most time consuming task.
c. Just do things and cross them off the list as she goes through her day.
d. Group them and then take care of them.

Figure 3.

Dictionary of Terminology representing the fundamental concepts of OOD.

Step 4. During the fourth step of the test construction process, the directions were created and administered by giving examinees verbal and written instructions to evaluate each question based *solely* on the information provided in the scenario. Each real-world scenario and question was designed to provide the necessary information for knowledge and comprehension in order for the examinee to accurately evaluate any given scenario. Each of the questions listed only one correct answer, two plausible distracters, and one obvious outlier.

Step 5. The test evaluation stage stresses that the test must be evaluated by a team of experts or by a focus group. Test construction experts from the Department of Education at Virginia Tech were consulted in conjunction with local OOD experts. Validity and reliability details will be discussed in the methods sections of this dissertation. The Design Readiness Assessment Scale (DRAS) was constructed and empirically tested, yielding internal consistency estimates of reliability ranging from .68 to .82. The current study aimed to reexamine the reliability and validity of the instrument.

Step 6. The study reported herein is the first attempt to evaluate the generalizability of the DRAS. A sample was drawn from two southeastern state universities. A report of the validity and reliability is discussed in the Method section. The DRAS will be used in conjunction with traditional measures of course performance to identify measures associated with OOD performance.

The following sections review the relevant literature, which is presented in relation to the dependent performance variables—course grade and design task. Significant critiques in the areas of objects-first programming, information systems, software engineering, and engineering design literature are also examined.

Course Performance Predictors

An enormous body of theoretical and empirical studies are readily available on the topic of predictors of course performance and achievement. In reviewing historical studies concerning predictors of computer science performance, there appears to be a growing list of explanatory variables. For example, studies before 1977 focused on demographic variables and high school achievement to account for enhanced computer science performance (Alspaugh, 1972; Denelsky & McKee, 1974; Peterson & Howe, 1975; Deckro & Woundenberg, 1977).Unfortunately, the models developed from the aforementioned studies demonstrated only limited predictive power (Evans & Simkins, 1989). Between 1977 and 1981, investigators broadened their research models to include the use of IBM's Programming Aptitude Test (PAT) (Mazlack, 1980; Fowler & Glorfeld, 1981; Stephens, Wileman & Kovina, 1981). However, the use of linear regression or factor analysis models in these studies failed to account for half of the total variation in course performance. Due to programming paradigm shifts, IBM's PAT soon became obsolete and researchers began to look for other possible predictors of success.

Since 1985, scholars have explored the relationships between computer science ability and general cognitive processes (Gibbs, 2000; Goold & Rimmer, 2000; Wilson, 2000; Cavaiani, 1989; Evans, 1989; Bishop-Clark, 1994; Werth, 1986). Although not every study reported significance in the use of cognitive measures, many of the surveys produced a wealth of valuable information on supporting the use of demographics and prior academic performance as possible indicators of computer science course success. Interestingly, these more recent reports validate some of the earlier studies prior to 1977 (Alspaugh, 1972; Denelsky & McKee, 1974; Peterson & Howe, 1975; Deckro & Woundenberg, 1977). More current reports have examined personality types and have found this variable to be only slightly significant in predicting computer science success (Capretz, 2002).

A sizeable number of the studies mentioned found correlations between their selection of independent variables and course performance. However, when several of these studies were replicated at other institutions, the results were inconclusive (Bishop-Clark, 1994; Gibbs 2000). Researchers have explained this outcome by characterizing the paradigm shift between procedural and object-oriented languages, variations in teaching styles, and/or the existence of confounding variables.

The following subsections will present a summary of related literature on the use of gender, prior computer science experience, cognitive abilities, and personality type as predictors of performance in introductory computing and engineering design courses. For this research, course performance is measured by grades on lab assignments, exams, and programming projects.

Gender

Prior to 1989, gender was a prominent predictor of course performance in first-year engineering design and computer science courses. One of the premier studies of computer science course performance prediction (Alspaugh, 1972) used gender as one of the ten variables of interest. Interestingly, the resulting data allowed for the explanation of 33 to 40 percent of the variation in course performance of 50 students. Similarly, Campbell and McCabe (1984) considered gender as a variable in their development of a linear discriminate model for computer science course performance. The use of gender allowed the researchers to successfully classify 175 of the 256 students (68.4%). Another notable study conducted by Jakiela and Fayad (1989) provided results that were in agreement with previous studies of gender performance prediction. Their study assessed 58 participants and found that male engineering design students were more likely than females to receive an 'A' in the course.

Since the early 90's, however, the performance outcomes in courses such as *introduction to procedural programming* and *introduction to objects-first programming* have found that gender has become an increasingly unreliable predictor of course performance. Similarly, the available literature on predictors of performance in introductory computer science courses using procedural languages shows that the gender gap is closing (Rountree, Rountree, & Robin, 2002; Goold & Rimmer, 2000, Wilson, 2000). Ventura (2004) further supports these finding in an objects-first java programming course, wherein gender was not significantly correlated with the course performance of 378 students.

In summary, while there are varying reports on the effectiveness of gender as a predictor of computer programming course performance, this research is primarily interested in exploring the association of gender with OOD course performance.

Prior Computer Science Experience

Prior computer science experience has been loosely defined as any experience/exposure to computing languages, equipment, and/or applications prior to the student's enrollment in the current course (Wilson, 2000). Jakiela and Fayad (1989) noted that prior computing experience was a significant predictor of course grade in an introductory engineering design course. When investigating programming language prediction, it was shown that prior BASIC knowledge was a positive predictor of performance on the second exam in the course (Evans & Simkins, 1989).

Hagan and Markham (2000) also examined the effect of prior programming experience on performance in an introductory Java programming course. They found that the students with prior programming experience performed better on the first and last stages of the course project (the second stage was omitted from their analysis), as well as on the first two exams and the overall course grade.

Lending and Kruck (2002) examined the use of prior programming experience in an introductory information systems course, but reported mixed findings. With over 300 participants in their study, they reported that prior computer science experience was not a significant predictor of course grade for female students, but was a significant predictor of performance for male students.

Studies involving introductory computer science courses have uniformly described that prior computer science experience was a significant predictor of course performance (Katz, Aronis, Allbriton, Wilson & Soffa, 2003; Morrison & Newman, 2001). In fact, Morrison and Newman not only showed that prior programming course experience was positively linked to introductory computer science course performance, but also demonstrated that it was particularly significant when that prior programming course was offered at the university level. Using a sample of 65 students, Katz et al. reported that the significance of prior computer science experience was only indirectly affected by gender since males generally had more prior computer science experience.

Two recent studies have reported findings contrary to the widely accepted belief that prior computer science experience is a reliable predictor of course performance (Ventura, 2004; Rountree, Rountree & Robin, 2002). Rountree et al. noted that only 18% of the students who claimed to already know a programming language had an appreciably higher success rate. They further noted that 12 of the 84 students who indicated prior experience actually failed the course. They concluded that knowing a programming language was no guarantee of course performance in an introductory computer science course. Ventura reported similar findings wherein a student's belief about his or her ability to program was not correlated with any measure of course performance. It was further noted that students without prior Java programming experience in fact did better than those who had programmed in Java prior to the objects-first course.

Due to the fact that there are no published studies regarding this effect for an OOD course, the current research will investigate what effect (if any) prior programming experience has on labs, exams, programming, and overall OOD course grade.

College Grade Point Average

There are varying reports of success in the use of grade point average (GPA) as an adequate predictive criterion in course performance relationships. Understandably, this inconsistency is to some extent due to the variation in courses taken by each student as s/he matriculates through high school and college. Furthermore, some students select easier external major courses than do other students within the same major. These observations suggest that GPA is probably a shifting, amorphous criterion, and therefore would be difficult to be used as a reliable predictor variable (Gall, Borg & Gall, 1996). Several studies were able to control for the variation in GPA by selecting a population with similar levels of academic exposure.

The performance outcomes in courses such as introductory engineering design, information systems, and computer science have corroborated that college grade point average is a significant predictor of course performance (Lending & Kruck, 2001; Jakiela & Fayad, 1989; Evans & Simkins, 1989). Evans and Simkins used stepwise linear regression models generated with grade point average as the first of three variable entered into the equation, which explained approximately 6.4 percent of the in variance programming assignments.

Jakiela and Fayad (1989) used 18 variables in an effort to predict course performance in an introductory engineering design course. They discovered that GPA was highly correlated (.89) with course grade. When entered in as the sole variable in a linear regression model, GPA explained 30% of the variance. However, for full regression runs only moderate multiple correlation squared (R^2) values were obtained with poor significance levels, leading the researchers to conclude that the utility of the final regression models was questionable.

Lending and Kruck (2001) found that the use of GPA in an introductory information systems course provided mixed results. Although male performance in the course could be reliably predicted using college GPA (among other measures), this variable was not significant in predicting female performance in the class.

In summary, the role of GPA as a predictor of course performance has been widely studied for related computing and engineering design course. These data will be used as a reference point for the experimental work of this dissertation.

Cognitive Abilities

According to Witkin, Oltman, Raskin and Karp (1971), cognitive abilities are the characteristic, self-consistent, modes of functioning that individuals show in their perceptual and intellectual activities. The following sections will present a survey of the literature on the various measures of cognitive abilities. Because of the limited amount of research available on cognitive abilities as predictors of performance in an object-oriented design, the literature related to computer programming and engineering design is also introduced. In summary, this research found that, spatial orientation, visualization, logical reasoning, field dependence/independence, and flexibility could all be considered potential cognitive abilities needed for OOD skills. OOD requires intuitive processing of decomposing complex situations into manageable chunks, and the aforementioned cognitive abilities are viable measures of this sort of mental processing.

Spatial Orientation

Ekstrom et al. (1979) defined spatial orientation is the ability to perceive spatial patterns or to maintain orientation with respect to objects in a space. Spatial orientation tests require participants to distinguish between the faces of an object. Each problem in the test features drawings of pairs of cubes or blocks. The second cube may be a rotated version of the first cube, and the subject has to determine if the two cubes are the same or different.

Visualization

Visualization is the ability to manipulate or transform the image of spatial patterns into other arrangements. The visualization and spatial orientation factors are similar but visualization requires that the figure be mentally restructured into components for manipulation, while the whole figure is manipulated in spatial orientation. Carroll (1974) concluded that both visualization and spatial orientation require the mental rotation of a spatial configuration in short-term visual memory; however, visualization requires the additional component of performing serial operations. Some subjects may employ an analytic strategy in visualization tests and search for symmetry and planes of reflection as clues to the solution.

Logical Reasoning

Logical reasoning reflects the ability to evaluate the logical correctness of possible conclusions for a given set of information. Ekstrom et al. (1979) defined logical reasoning as the ability to reason from premise to conclusion, or to evaluate the correctness of a conclusion. The complexity of this factor has been pointed out by Carroll (1974), who describes it as involving both the retrieval of meanings and of algorithms from long-term memory and then performing serial operations on the materials retrieved. Carroll suggested that individual differences on this factor could be related not only to the content and temporal aspects of these operations, but also to the attention that the subject gives to details of the stimulus materials.

Flexibility

Flexibility of use is the mental mindset necessary to think of different uses for objects. Flexibility tests typically ask the subject to practice "practical resourcefulness" in naming two objects that can be used together to make something or do something that is required (to solve a particular task).

Jakiela and Fayad (1989) conducted a study using 21 factors that were thought to contribute to engineering design skills—one of which was a measure of flexibility. Although they reported that flexibility was not a significant variable in the overall analysis of the data, there was a significant correlation between gender, flexibility, and overall performance. Moreover, Jakiela and Fayad found that females were generally more "flexible" in their thinking and in the application of the problem domain, which ultimately resulted in higher success rates. Those females that were found to be more *flexible* performed better in the course, as opposed to the less flexible-thinking females.

The Factor Referenced Kit of Cognitive Tests

The Kit of Factor-Referenced Cognitive Tests contains tests validated in previous studies. Reliable versions of the aforementioned tests are found in the factor reference kit of cognitive tests.

Mayer, Dyck and Vilberg (1986) conducted a study using 111 computer-naïve computer science students. They administered measures of spatial ability, logical reasoning, visual ability, verbal ability, and a host of other potential explanatory variables. Mayer et al. found that tests measuring logical reasoning were significantly correlated with learning a programming language ($r = 0.54$). In addition, tests of spatial ability were significantly correlated with college students learning Logo programming ($r = 0.49$).

Scanlan (1988) conducted a study using the kit of Factor-Referenced Cognitive Tests (Ekstrom, French & Harman, 1979) to assess mental aptitude associated with the programming ability of 45 subjects. They used 18 tests to measure separate cognitive abilities. Factor analysis was used to analyze the programming grades in an introductory-level course along the 18 cognitive factors. Of these tests, seven were found to account for approximately 60% of the variance in programming aptitude. Logical reasoning and verbal comprehension were reported as loading at $r = .61$ with ($p < .0001$) and flexibility of use is reported as loading at $r = .41$ with ($p <.01$). Scanlan also found that these cognitive tests were useful for developing equations using stepwise regression analysis.

Allen (1992) used the kit of Factor-Referenced Cognitive Tests (Ekstrom, French & Harman, 1979) to study the effects of cognitive ability on end user searching of a CD-ROM Index, using a sample size of 50 students. Allen utilized tests for measuring verbal comprehension, perceptual speed, spatial scanning and logical reasoning. Allen confirmed that cognitive abilities did affect a participant's ability to retrieve information. Participants with low scores in logical reasoning were less selective in identifying the necessary information.

Perceptual Style

The final cognitive ability of interest within this research is perceptual style. Perceptual style is the manner in which a person cognitively approaches a learning situation. In terms of perceptual style, an individual can be classified as field-independent or field-dependent. Field-dependence/independence (FD/I) is the most extensively researched cognitive control. Initiated over 40 years ago, FD/I remains among the most prescriptive of learning and instructional outcomes (Grabowski & Jonassen, 1993). Stevens, Wileman, and Konvalina (1981) and Werth (1986) are particularly renowned for using measures of FD/I as reliable predictors of success in computer science courses.

Individuals who prefer a field-dependent (FD) perceptual style tend to perceive globally, i.e., perception is dominated by the overall organization of the surrounding field, and parts of the filed are experienced as "fused" (Witkin et al., 1971). In a field-dependent mode of perception, individuals have greater difficulty solving problems, are more attuned to their social environment, learn better when concepts are humanized, and tend to favor a spectator approach to learning. Additionally, individuals preferring a field-dependent perceptual style have been found to be more extrinsically motivated and prefer that organization and structure for the subject matter be provided by the teacher (Witkin1977).

Individuals who prefer a field-independent (FI) perceptual style tend to view concepts more analytically, finding it easier to solve problems. Specifically, field-independence is characterized by an individual's success at separating relevant material from its context—in other words, being able to discern the *signal* (the relevant) from the *noise* (the incidental from the peripheral). Moreover, these individuals are more likely to favor learning activities that require individual effort and study. They prefer to develop their own structure and organization for learning, are intrinsically motivated, and are less receptive to social reinforcement (Witkin, Moore, Goodenough & Cox, 1977).

Several tests can be used to classify an individual's perceptual style. The following section will briefly discuss these tests and will present a detailed description of the most reliable and widely used test, the Group Embedded Figures Test (GEFT), developed by Witkin, Oltman and Raskin (1971).

The Group Embedded Figures Test

The GEFT is a standardized instrument that has been used in educational research for more than 25 years, and was designed to establish whether an individual's perceptual style could be classified as field-independent or field-dependent. The respondent is typically asked to identify eighteen simple forms hidden within complex figures with scores ranging from zero (field dependency) to 18 (field independency). As described earlier, field-independent individuals are able to perceive items as separate from a surrounding field. Conversely, the perception of field-dependent individuals is strongly dominated by the surrounding field—e.g., such individuals would be unable to accurately adjust a test rod to its true vertical and would experience difficulty discerning geometric shapes and patterns from complex designs.

The validity of the GEFT has been established by significant positive correlations with the individually administered *Embedded Figures Test,* as well as other instruments, such as the *Rod and Frame Test*, designed to measure like constructs. Acceptable reliability scores (r = .82) have been demonstrated (Witkin, Oltman, Raskin & Karp, 1971), indicating that males tend to score noticeably higher (p < .005) than females. This is consistent with the literature suggesting that males generally have higher levels of field independence than females (Witkin, Oltman, Raskin & Karp, 1971). Studies investigating cognitive style and race have shown mixed results (Kush, 1996; Shade, 1981).

Despite its widespread use for over two decades, the GEFT has been somewhat inconsistent in measuring conceptual development. Nonetheless, this research explores its association with performance in an OOD course.

Currently, there are no studies using a battery of test results to measure/predict OOD performance. This research, therefore, will explore the use of the kit of factor referenced cognitive test (measuring spatial ability, visualization, logical reasoning, flexibility of use) and the Group Embedded Figures Test as indicators of performance in an OOD course.

Moreover, this research will represent an exploratory analysis of the variables associated with OOD course performance as measured by lab grades, programming assignments, exams, and overall course grades. While course grades are generally considered to be accurate measures of performance and are directly related to course goals and objectives; they can also be subjective and may not provide dependable information on how performance relates to educational experiences.

The following section will discuss the literature relevant to the use of a design task and a design assessment rubric as objective measures of performance.

Design Task Performance

The ACM/IEEE Joint Task Force on Computing (CC2001) noted that object-oriented programming *and* design should be included in introductory computer science courses—if the computer science program were to follow an objects-first curriculum model. However, assessing the learning that results *only* from the design portion of the course can be difficult to operationalize or measure, and traditional assessment tools or existing instruments may not be appropriate (Athman, Adam & Turns, 2000). While design may be the focus of some assignments and/or tests, a resulting course grade can also be impacted by other factors (i.e., programming assignments, quizzes, homework, and attendance). This research concluded that a consistent measure of OOD performance should be investigated.

To gain insight into the theory and application of design performance measures, research on the use a design task was evaluated. Studies involving students in the areas of computer science, software engineering, and engineering design will be introduced in the following paragraphs.

Factors Associated with Design Performance

Jakiela and Fayad (1989) conducted a study consisting of 58 engineering design students. The students were given a "kit" of various scrap parts and were asked to construct a machine. The conceptualization, design, building notes, and any other pertinent information were kept in a lab notebook. The notebooks, along with the craftsmanship of their submitted machine, were used to gauge performance. Subsequently, Jakiela and Fayad measured 18 factors that were thought to be related to design performance. According to their tests of cognitive abilities, women were identified as more flexible designers. They also noted an interesting point of discussion, "… because female students were more flexible in their overall designs, the outcome of the design may not have been what the (less flexible) male professor was looking to assess, but it does not necessarily mean that their design was any less correct" (p. 301). However, the cognitive measure of flexibility was not found to be significantly associated with design task performance. Using linear regression analysis, Jakiela and Fayad concluded that academic class, college GPA, home income, and perceived competency were associated design task performance.

The design tasks were not evaluated by formal grading schema. Instead, designs were appraised on their success in a machine vs. machine contest. Individuals were evaluated on the construction quality of their machine and the preparedness of the machine operator on the day of the contest.

In the field of software engineering education, McCracken (2002) developed and tested a model consisting of three levels of design skills: (1) meta-cognitive skills and domain independent knowledge, (2) domain specific design skills knowledge, and (3) domain-specific problem-solving skills. This model was based on prior research published in the areas of architecture, software, and mechanical engineering.

The researcher tested his model by conducting a study involving three participants with varying levels of OOD experience—from expert to relative novice. McCracken instructed participants on the think-aloud protocol that was to be used to gather information on the development of an Automated Teller Machine (ATM) design task. The design sessions were not timed, but generally took between 1 and 1½ hours to complete. The think-aloud protocol was coded similar to a schema published in engineering education (Atman & Bursic, 1998). The coding was not tested for inter-rater reliability. McCracken's design model enabled researchers to predict the behavior of the expert, but more importantly, it also allowed them to explain the specific learning issues that impacted the test subjects' behaviors.

While McCracken's research is a notable advance in the development of a model for designing, the study lacked the adequate sample size to be able to generalize the results. Furthermore, the model lacked clear guidelines on the transition between the levels, as well as what salient skills were covered in each level.

Design Task Assessment

Gentili et al. (1999) conducted a study to assess design capabilities of students enrolled in an introductory engineering design course. Specifically, the researchers developed assessment methods to determine students' design capabilities prior to and at the mid-point of an engineering course. Four different assessment methods were used: short-answer exam, team design assignment, reflective paper, and self-assessment.

The results of the study revealed that the combination of these assessments consistently and predictably revealed student progress. The researchers stated that a course structure designed with these assessment strategies in mind could produce measurable gains in student skills and knowledge in engineering design. Reliability and validity of the assessment instrument were not included in the published article.

Custer, Valesey, and Burke (2001) developed and tested an assessment model designed to measure student problem-solving performance in technological design activities. A rubric incorporating critical incidents in problem-solving and expertise levels was central to the model, which was intended to provide a framework for assessing technological problem-solving in group and individual activities. The research further sought to identify specific factors (i.e., GPA, grade level, technology courses, mathematics and science grades, gender, personality preferences, and problem-solving styles) that were associated with design activities in high school students.

The *Student Individualized Performance* (SIP) rubric was developed to assess individual student performance in technological problem-solving situations. Based on a synthesis of the design literature, the researchers identified four major dimensions that were consistently represented in various design and problem-solving models: (1) Problem & Design Clarification, (2) Develop a Design, (3) Model/Prototype, and (4) Evaluate the Design Solution. Each dimension was subdivided into three strands, replicating the process used to identify the major dimensions. These dimensional categories were reviewed by an expert panel with extensive knowledge of problem-solving and design for conceptual accuracy.

The researchers conducted an orientation session consisting of a brief discussion of design and problem-solving, a verbal description of the design brief, and a period of clarification and discussion. Students were asked to engage in a process of design clarification, design development, physical modeling, and evaluation. Each student was issued an actual school locker and materials (i.e., markers, foam board, tape, scissors, cardboard, and hot glue guns) to use to construct a full-size mock-up of their design. The designs were evaluated using the SIP rubric. The inter-rater reliability for the SIP rubric was reported as .78.

Custer, Valesey, and Burke (2001) concluded that the investigation of factors associated with design activities was largely exploratory in nature; thus only descriptive statistics were presented. These correlation results suggested definite relationships between technological design performance, GPA, and science achievement.

Summary

As suggested by the literature, designers attack problems by decomposing them into solvable sub-problems and rely on previous experiences to guide them in the design process (Coplien, 1996). As noted earlier, students are having difficulty learning and applying OOD strategies, which can be explained by (1) the varying conceptualizations of OOD, and (2) their readiness to learn OOD. An investigation of OOD literature resulted in the identification of seven fundamental OOD principles/strategies. The OOD principles were then introduced into a theoretical model of OOD problem-solving, adapted from Polya's mathematical problem-solving model. This model will be used to create an instrument of design readiness.

This research also investigated the characteristics of a student's background and/or cognitive ability that could be associated with OOD course performance and design task performance. The notion of gender and prior computer science experience, as measures of performance ability, have been well documented for introductory computing and engineering design courses. Although gender has not been shown to be a significant measure of computer science course performance since 1989, this research will explore gender role in the performance in an OOD course and on a specific OOD task.

Prior computer science experience has been shown to have a strong positive association with enhanced performance in introductory procedural programming courses. Conversely, prior computer science experience seemed to have a negative association with course performance in an objects-first programming course. Therefore, this research was designed to explore the relationship between prior computer science experience, OOD course performance, and OOD task performance.

Cognitive factors were also presented with promising results regarding their association with course performance. The ETS Kit of Factor Referenced Cognitive Test and the Group Embedded Figures test have shown reliability in their measurement and prediction of course performance. It is the expectation of this investigator that these tests will prove equally reliable when measuring OOD performance.

The research introduced throughout this chapter used the term “predictors” rather loosely. While this study endeavors to provide analytic data to support strong associations between various criteria and course performance, many investigators have made assertions about predictors without a theoretical framework on which to base their findings (i.e., Ventura, 2004; Rountree, Rountree, Robin, 2002; Jakiela & Fayad, 1989). While examining the literature,this investigator was only able to identify a limited number of studies that actually used previously reported models as their theoretical foundation (i.e., Lending & Kruck, 2002; Gibbs, 2000; Evans & Simkins, 1989).

Because of the lack of predictive literature in OOD, this study makes no assertions as to the predictors of performance in OOD. Rather, this research merely seeks to identify measures that are associated with OOD in anticipation of developing an OOD prediction model in future work. The references to related subject matter will be used as comparison points for findings associated with OOD course performance and design task performance within this research.

CHAPTER THREE

METHOD

The purpose of this study was to identify factors associated with OOD performance. Two distinct variables, performance on the post-training design task, and overall grade in an OOD course, were used to measure OOD performance. This study investigated factors such as demographics, prior computer science experience, personality type, prior academic success, and cognitive abilities. Self-reporting, standardized tests, questionnaires, and observational techniques were used to determine the aforementioned predictor variables. In addition, this study introduced a novel measure of design readiness to be used in mediator analysis as a procedure for improving the predictability of OOD performance. Finally, path analysis was used to identify associations between variables.

Sample

Students enrolled in CS2 (see definition in terminology: Chapter 1 – Terminology) courses at two southeastern state institutions were asked to participate in this study. Grades were obtained from the course instructors. The duration of the data collection was one semester (January through May, 2004).

Based on student population categories as reported in *U.S. News and World Report* (2004), School A was selected from what was considered to be a medium-sized university (~16,500 students). School B was selected from the list of larger state universities (~25,000 students). All participants were enrolled in a CS2 course during the spring semester of 2004. Both universities follow the IEEE/ACM Joint Task force on Computing Curriculum (CC2001) syllabus for object-first pedagogy. Specifically, CC2001 stated that a CS2 course should cover topics related to the introduction of object-oriented programming and design. The syllabus for each course is found in Appendix B – Course Syllabi.

Solicitation for participation in this project was conducted via e-mail and through in-class visitations. Students were offered the incentive of opting out of the last course project by participating this in this study, which resulted in an average participation rate of 90%. Several of the students (n = 5) who decided not to participate were briefly interviewed, and 80% of these non-participants (n = 4) reported they enjoyed programming more than design, so the incentive offered did not appeal to them.

The overall mortality rate experienced during this study was 22% (n = 45). As shown in Table 2, the number of participants from School A decreased by 25% (n = 25). When examining the results, both sections within Sample 1 experienced an almost even loss of men (n = 12; n = 11). The results showed no loss of women in the first section and a 16% (n = 2) loss of women from the second section of the CS2 course.

Table 2 also details the 19% loss (n = 20) of participants from School B. While there was only a 15% loss of men from this sample, there was a 71% (n = 5) decrease in women participants. The results show a loss of over half of the women enrolled in both sections of the CS2 course.

The drastic reduction in women participants from School B was somewhat problematic for this research, as gender was hypothesized to be positively associated with OOD success. Measures were taken to ensure that a representative sample of men, possessing similar characteristics to those of the remaining women, was identified to test this hypothesis.

Careful consideration was also taken when selecting the number of variables used in data analysis, since sample size was a matter of concern. With the current sample size (n = 161), the introduction of a large number of variables increased the likelihood that the predication variance could be merely attributed to chance (Gall, Borg & Gall, 1997).

Table 2.

The Effects of Participation Mortality

SCHOOL A

	COURSE LECTURE SECTION		TOTAL
	9:00 a.m.	1:00 p.m.	
MEN			
Initial	50	39	89
Final	38	28	66
WOMEN			
Initial	4	8	12
Final	4	6	10
TOTAL			
Initial	54	47	101
Final	42	34	76

SCHOOL B

	COURSE LECTURE SECTION		TOTAL
	9:00 a.m.	11:00 a.m.	
MEN			
Initial	34	64	98
Final	28	55	83
WOMEN			
Initial	3	4	7
Final	1	1	2
TOTAL			
Initial	37	68	105
Final	29	56	85

Measures

Given the paucity of measures predicting OOD readiness or successful performance, this study selected variables traditionally used to predict computer programming ability. Specifically, this research administered measures of prior computer science experience, cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility of use, and perceptual style), and design readiness (Design Readiness Assessment Scale). The final measure used in this study was a specially constructed pre/post training design task.

Each variable was assessed using formats to be described in subsequent sections. Responses to the respective variable statements were tallied to generate a total score based on correct/incorrect responses to specific items.

Prior Computer Science Experience Scale

The prior computer science experience scale was specially constructed for this research. Participants responded based on their self-assessed proficiency in various areas of computer science. Five subscales were identified (UNIX Programming, Object-Oriented Processes, Web Designing, Computing Platforms, and Various CS), each of which was named according to the majority of the items represented by the subscale (Green & Salkind, 2003). The prior computer science experience scale can be found in the background survey in Appendix C – Background Survey.

Participants responded to the prior computer science experience scale using a 5-point Likert-type scale, 1 (None), 2 (Novice), 3 (Intermediate), 4 (Proficient), and 5 (Expert). The use of a 5-point scale is supported through existing research suggesting that coefficient alpha reliabilities tend to increase up to the use of 5-points and level off thereafter (Hinkins, 1995). Each subscale within the prior computer science experience was determined by calculating the average of the sum of the proficiency levels for each item.

The UNIX Programming experience scale consists of six items: PERL, UNIX, JSP, CGI, TCL_TK, and ASP. The Object-Oriented Processes experience scale encompasses five items: OOP, OOD, LARGE_PROGRAMMING, C++, and JAVA. The Web Designing experience scale includes five items: WEB_DESIGN, DREAM_WEAVER, HTML, FRONT_PAGE, FLASH. The Computing Platform experience scale consists of six items: NETWORK_PROTOCOL, LAPTOP_USE, WIRELESS_NETWORK, DOS, WINDOWS, and MAC. And finally, the Various CS experience scale includes four items: UML, TEAM_WORK, DATABASE_PROGRAMMING, and VISUAL_BASIC. A high score on the scale designates a high level of computing experience.

Cognitive Abilities

The following five measures of cognitive abilities were selected for this study, each of which was explored as a skill that might predict performance in OOD.

Spatial Orientation

Spatial Orientation was assessed using the Cube Comparison Test – S-2, available through the Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, & Harman, 1979). The cube comparison test features cubes with a different letter, number, or symbol on each of the faces. Each problem in the test consists of selecting pairs of cubes. The second cube may be a rotated version of the first cube. The participant has to determine if the two cubes are the same or different. Participants are instructed not to guess because their score is determined by the number correct minus the number incorrect. Prior studies identified within the Kit of Factor-

Referenced Cognitive Tests (Ekstrom, French, Harman, and Dermen, 1976) reported reliabilities ranging from .77 to .84.

Visualization

Visualization was assessed using the Surface Development Test – VZ -3, available through the Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, & Harman, 1979). The participant is asked to imagine or visualize how a piece of paper could be folded to form a particular object. There are two drawings that the participant must consider. The drawing on the left is a piece of paper that must be folded on the dotted lines to form the object drawn on the right. The participant has to identify which of the lettered edges on the folded object is the same as the numbered edges on the original object. An instructional note states that the flat piece of paper marked with the X will always be the same as the side of the object marked with the X. The paper will always be folded so that the X will be on the outside of the object. Participants are instructed not to guess because the number correct minus the number incorrect determines their score. Prior studies identified within the Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, Harman, and Dermen, 1976) reported reliabilities ranging from .75 to .94.

Logical Reasoning

Logical reasoning was assessed using the Inference Test - RL-3, available through the Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, & Harman, 1979). Each item on the Inference Test requires the student to read one or two statements that might appear in a newspaper or popular magazine. A participant is asked to select only one of five statements representing the *most* correct conclusion that could be drawn from the available statements. The student is instructed not to consider information that is not given in the initial statement(s) in order to draw the most correct conclusion. The participant is also advised not to guess, unless he or she can eliminate possible answers to improve the chance of choosing, since incorrectly chosen responses will count against him/her. Prior studies identified within the Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, Harman, and Dermen, 1976) reported reliabilities .57 to .78.

Flexibility of Use

Flexibility of use was assessed using the Combining Objects Test – XU-1, available through the Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, & Harman, 1979). Participant is asked to demonstrate "practical resourcefulness" by naming two objects that could be used together to make or do something specific (to solve a particular task). The participant must name objects that are found in the specified locations. Each task in this test will indicate the location of the object, as well as which object(s) might be lacking. The participant is required to name two objects that would usually be found in the given location and which could be used together to fulfill the request. The number of correct responses determines the final score. A prior study identified within the Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, Harman, and Dermen, 1976) reported a reliability score of .80.

Perceptual Style

Perceptual style was measured by the Group Embedded Figures Test – GEFT (Witkin, Oltman, Raskin & Karp, 1971). Participants were asked to identify a number of simple structures embedded within a complex figure. This measure consisted of 18 items of various geometrical shapes and shadings. A higher score represents a trait commonly known as *field-independence*, which was described earlier. A lower score corresponds to field-dependence. This measure is consistently reported (e.g., Werth, 1986; Chamillard & Karolick, 1999) as having an internal consistency reliability of .82.

The Design Readiness Assessment Scale

Design Readiness was assessed using the Design Readiness Assessment Scale (DRAS) developed and validated by Lewis, Pérez-Quiñones, and Rosson (2004). The DRAS assesses one's understanding of OOD concepts and strategies. The participant is given twelve problem scenarios covering OOD techniques: divide and conquer, encapsulation, information hiding, inheritance and polymorphism, generality, and abstraction. Participants were instructed to read each scenario and reflect on the interaction between people, events, and objects. They were further instructed *not* to read "real-life" experiences in the scenarios, but to choose the best answer to each question based solely on what is stated in the scenarios. Correct responses were tallied to generate a final score. Pilot studies show prior reliabilities between .68 and .82.

Pre/Post Training Design Task

The pre/post training design task asked participants to create a multi-player "survival of the fittest" terrarium game. The participants were asked to identify the essential classes and interactions between those classes. Each participant was instructed that s/he was not required to *code* any of the specifications of the game, however, they were required to *design* the game in such a way that a coder could advance from their design directly into coding of the system.

The design task was conducted following the one-group pre-test/post-test research procedures detailed by Gall, Borg, and Ball (1997): (1) administration of the design task measuring pre-training design ability; (2) course instruction; and (3) administration of the same design task measuring post-training design ability. The effect of knowledge gained during course instruction was determined by comparing the pre-training and post-training scores.

The design tasks were evaluated using an adapted version of the Student Individualized Performance (SIP) rubric developed Custer, Valsey, and Burke (2001). The SIP consists of six categories: cohesion, clarity, completeness, clarity, consistency, and correctness. Each category was evaluated using a 5-point Likert-type scale, 1(Novice) to 5(Expert). The category points were totaled to suggest an overall score on the design task. Criteria for each category and evaluation point varied across the SIP.

Three evaluators—two external evaluators and the primary investigator of this research—were trained to assess the design tasks using the SIP. Prior research (Custer, Valsey, Burke, 2001) reported inter-rater reliabilities between .50 and .78. These researchers attributed the lower reliabilities to inadequate training of the evaluators. To improve the likelihood of higher inter-rater reliability, the external evaluators used in this study were trained according to the guidelines established by Custer et al.

The principal researcher of this study conducted a half-day training workshop during which external evaluators were instructed on the use of the SIP. Each evaluator was asked to complete the design task prior to the training workshop. The training workshop consisted of an SIP orientation session, design evaluation sessions, and concluded with an open discussion on design evaluation techniques. During the orientation session, evaluators were instructed on the *minimum* requirements for the design task solutions and then instructed as to how these requirements were mapped to the SIP. For this study inter-rater reliabilities ranged from .81 to .83.

Data Collection Procedures

Institutional Review Board approval was obtained from both universities prior to data collection. This research was granted IRB exemption from one university and expedited review process from the other university. Study participants from both universities were required to read an online version of the informed consent form before participating in this study. All participants were assured that their responses would be kept anonymous, and no personal identifiable information would be published with the results of this research. The consent form is found in the appendix of this document.

The data collection took place at three intervals between January and May of 2004. The first data collection point gathered general demographics and three measures of cognitive abilities (spatial orientation, visualization, perceptual style) used in this study. During the second data collection point the DRAS, two measures of cognitive abilities (logical reasoning and flexibility), and the pre-training design task were administered. The final data collection period involved the administration of the post-training design task. The universities involved in this study were staggered by one week in their spring 2004 academic schedule, allowing minimum overlap in data collection points 1 and 3. While the primary investigator of this research conducted 95% (n = 196) of the design laboratory sessions, a design laboratory assistant was hired and trained to conduct a small number of the design laboratory sessions. The design laboratory administration guidelines are included in the Appendices. The data collection timeline is outlined in Table 3.

In order obtain an accurate measurement of pre-training design ability, it was critical that the administration of the pre-training design task occur as early as possible in the academic semester. Correspondingly, the post-training design task was conducted during the final week of the semester to maximize exposure to design training. The maximum time between design laboratory sessions was approximately 3 months. Prior research involving predictors of object-first success (Ventura, 2004) concluded that 3 months was an appropriate time frame to assess knowledge gained in a one-semester college course.

Table 3.

Design Readiness Study Data Collection Timeline

Time in Semester	Location	N	Instruments	Time Required
First week of Classes				
1st day of class	In-Class	206	Introduction of Dissertation Research	10 minutes
2nd day of class	In-Class	206	Participant Background Survey Group Embedded Figures Test Spatial Orientation Test Visualization Test	55 minutes
First month of classes (student selects time and day)	Design Laboratory	206	Myers-Briggs Type Indicator Design Readiness Scale Logical Reasoning Flexibility Test Pre-Training Design Task	1 hour, 30 minutes
Final week of classes (student selects time and day)	Design Laboratory	161	Post-Training Design Task	1 hour

Analysis

To explore the relationship among prior computer science, cognitive abilities, design readiness, and OOD performance, this research utilized *path analysis* in combination with other multivariate methods (multiple regression, correlation, and factor analysis) because of its predictive power in exploratory studies (Gall et. al, 1997). According to Gall, Borg, and Gall, path analysis consists of four steps: (1) formulating hypotheses that causally link the variables of interest, (2) selecting or developing measures (theoretical constructs) of the variables that are specified in the hypotheses, (3) computing statistics that show the strength of relationship between each pair of variables that are causally linked in the hypotheses, and (4) interpreting the statistics to determine whether they support or refute the theory. The quantitative data was analyzed using SPSS 12.0.

Methodological Assumptions

Path analysis is a sensitive technique that must be free of confounding conditions in order for it to yield meaningful results (Gall, Borg, & Gall, 1997). Specifically, results could be misleading if (1) variables are not well measured, (2) important causal variables might be left out of the theoretical model, or (3) the risk of multicollinearity may be present. Furthermore, the sample size may be insufficient for the number of variables being considered (Chalikia, 2000; Gall et. al, 1997).

Path Model Analysis

This research used recursive path models that considered only unidirectional causal relationships. The use of path models typically requires hypothesized causal links between variables. However, due to the exploratory nature of this study, no predictive assumptions were drawn from the path analysis. Instead, this path analysis was simply used as a relationship explanatory tool.

CHAPTER FOUR

RESULTS

As described in Chapter 1, the primary goals of this research were to (1) develop an OOD problem-solving model, outlining the progressive stages and principles needed for teaching and learning OOD; (2) construct and validate an instrument designed to measure object-oriented design readiness, namely, the Design Readiness Assessment Scale (DRAS); and (3) identify experiences and cognitive measures that are associated with object-oriented design performance.

The results of this study are presented in three sections. First, a description of the sample is provided in terms of the continuous variables of interest for the study. These include prior computer science experience, cognitive abilities, design readiness, and course grade. This section will also discuss the demographic representations of the sample, reliability and factor analyses, and interpretation of the mean differences between schools for the measures of individual differences. The individual differences measures are defined as prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), and cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility and perceptual style). *Multivariate analysis of variance* (MANOVA) and independent-samples *t*-test were used to determine whether the two schools used in this study differed on more than one dependent variable.

Section Two assessed the study's five research questions using multiple regression analysis. Particular attention was given to the degree to which the various demographic and cognitive abilities variables were associated with design readiness, post-training design task score, and OOD course grade.

Two frequently scrutinized student characteristics (gender and college grade point average) were considered for entry into the regression models as control variables. In computer science prediction literature, however, recent studies have shown that gender is not significantly associated with course performance (Ventura, 2004; Rountree, Rountree, & Robin, 2002; Wilson & Soffa 2001; Goold & Rimmer, 2000). Although this investigator believed that this would hold true in the current analysis as well, too few women were represented in the sample to be able to include gender as a control variable in the regression analysis. Instead, college grade point average was entered as a control variable because it tended to be the basis of many performance prediction models (Lending & Kruck, 2002; Katz, Aronis, & Allbriton, 2001; Eskew & Faley, 1999; Evans & Simkins, 1989).

The final section presents exploratory path models, illustrating how performance on a design task from pre- to post-training is associated with background characteristics and design readiness. The chapter concludes with a summary of the research questions addressed in this study.

Descriptive Statistics

This study included 161 students enrolled in the CS2 course from two universities located in the southeastern region of the United States. In an effort to combine the schools into one sample, the investigator analyzed the degree to which the two schools were similar.

Demographic Analysis – School A

This sample was 87% men, with a mean age of 19.5 (SD = 1.36). The mean high school grade point average was 2.93 (SD = .38), and the mean college grade point average was 2.83 (SD = .38). A substantial portion (95%) of the participants was enrolled as full-time students and 91% reported English as their native language. Thirty-eight percent were classified as freshmen, 25%

as sophomore, 22% as junior, and 15% as senior. Thirty-seven percent reported being computer science majors, and 23% reported being information systems majors. More than half of this group (61%) reported taking one or more computer science courses in high school, 23% took one or more advancement placement computer science courses in high school, and 10% took the advanced placement computer science examination.

Demographic Analysis – School B

This sample was 98% men, with a mean age of 18 (SD = .68). The mean high school grade point average was 3.56 (SD = .39), and the mean college grade point average was 2.95 (SD = .35). Nearly the entire sample (98%) was enrolled as full-time students and 96% reported English as their native language. Eighty-six percent of the students were classified as freshman, 11% as sophomore, and 2% as junior. Ninety-six percent reported being computer science majors. A majority (81%) reported taking one or more computer science courses in high school, 39% took one or more advanced placement computer science courses in high school, and 29% took the advanced placement computer science examination.

Demographic Analysis – School Comparison

Chi-squares and Independent-sample *t* tests were conducted to evaluate the differences between the schools using demographic variables previously reported (age, high school grade point average, college grade point average, enrollment status, native language, classification, major). School A (men – 87%; women 13%) had significantly more women than did School B (men – 98%; women – 2%); $X^2(1) = 6.79$; $p < .01$. It was found that the two schools significantly differed in the means of age ($t(159) = 6.63$, $p = .00$), high school grade point average ($t(159) = -9.72$, $p = .00$), classification ($t(159) = 7.68$, $p = .01$), and advanced placement courses taken (*t*(159) = -2.40, *p* = .02). Based on the significant differences in the demographic means, the variable *school* was introduced into the regression model as a control variable.

Reliability and Factor Analyses

The Cronbach's alpha for the previously validated measures of cognitive ability—spatial orientation (α = .92), visualization (α = .94), flexibility (α = .85), logical reasoning test (α = .69), and perceptual style (α = .84)—were within acceptable ranges established in previous studies of college students (Ekstrom, French, Harman, with Dermen, 1976; Witkin, Oltman, Raskin & Karp, 1971). According to Peterson's (1994) meta-analysis of coefficient alphas, an alpha value of .70 is recommended for psychological constructs. Therefore, careful consideration was taken with the use of the logical reasoning test (α = .69), as it is slightly below the minimum acceptable coefficient alpha.

Factor analysis was conducted on the prior computer science experience scale, as well as the design readiness assessment scale. Weiss (1970, p. 478) recommended factor analysis for those situations where investigators wished to "reduce their variables to a smaller set by essentially decomposing the original variables into a new subset of variables composed of linear combinations of parts of the variance of the original variables."

Prior Computer Science Experience Scale

The prior computer science experience scale (PCSES) was specifically constructed for this study. Factors were analyzed using principal components analysis and Kaiser's normalized varimax orthogonal rotation. Employment of Kaiser's criterion of factor acceptability (associated eigenvalue greater than one) yielded six factors. Solutions of lower dimensionality and scree plots were examined and a five-factor solution was determined to be most meaningful. Item loadings on these five factors are shown in Table 4. The factors were interpreted by examination of the variables according to three guidelines: (1) four or more measures would be chosen to represent each construct of interest (Green & Salkind, 2003), (2) items that were cross loading or items where the factor loading between .40 and -.40 would be deleted (Nunnally, 1978), and (3) each underlying factor should be interpretable (Green & Salkind, 2003). Table 4 shows the subscales that were created based on principal component extraction forced to five factors.

The first factor, which accounted for 23.58% of the variance, was entitled as UNIX programming. The second factor, which accounted for 7.94% of the variance, was entitled as object-oriented processes. The third factor, which accounted for 6.78% of the variance, was entitled as web designing. The fourth factor, which accounted for 5.85% of the variance, was entitled as computing platforms. And finally, the fifth factor, which accounted for 5.03% of the variance, was entitled as various CS tasks.

Internal consistency estimates were computed for the five subscales within the prior computer science experience scale. The coefficient alpha for UNIX programming knowledge (α = .76), object-oriented processes (α = .78), web designing (α = .84), and computing platforms (α = .71) are modestly above the acceptable internal consistency value (.70) for scales (Nunnally, 1978). However, the various CS subscale (α = .27) was found to be below the recommended minimum standard. The various CS subscale was included in further analyses, with the caveat that any significant relationships found with this scale would not be applicable beyond the current study.

Table 4.

Principal Component Extraction, Varimax Rotation with Kaiser Normalization – Factor Analysis of Prior Computer Science Experience Scale

	Component					Communalities
	1	2	3	4	5	
PERL	**.80**	.16	.20	-.05	-.02	.70
UNIX	**.72**	-.06	-.11	.30	.22	.68
JSP	**.68**	.17	.10	.02	-.02	.50
CGI	**.67**	.19	.36	.01	.08	.62
LINUX	.67	.07	-.00	.43	.07	.64
TCL_TK	**.58**	.11	-.09	.07	-.11	.37
ASP	**.56**	-.04	.40	.16	.26	.57
JAVA_SCR	.47	.09	.41	.07	.26	.47
DOT_NET	.36	.30	.29	-.02	.29	.39
ADA	.31	-.05	.19	-.03	.27	.21
OOP	.02	**.79**	.14	.09	.12	.67
OOD	.04	**.74**	.11	.11	.18	.61
LG_PROG	.05	**.72**	.04	.20	.07	.56
C_PLUSPL	.05	**.69**	.12	.05	-.39	.64
C	.40	**.65**	.07	.16	-.13	.63
RLTM_SYS	.11	.58	.08	.21	.41	.56
JAVA	.21	**.48**	.21	.01	.08	.33
WEB_DSGN	.07	.13	**.76**	.37	-.04	.74
DRM_WVR	.14	.14	**.74**	.20	.02	.63
HTML	.25	.14	**.71**	.27	-.03	.66
FRONTPG	.02	.05	**.69**	.02	.25	.55
FLASH	.06	.30	**.68**	.20	.09	.60
NET_PROT	.14	.14	.08	**.70**	.01	.54
LAPTOP	-.01	.03	.09	**.68**	.19	.50
WLS_NET	.07	.19	.11	**.66**	.26	.56
DOS	.17	.19	.16	**.60**	-.03	.45
WINDOWS	-.01	.09	.20	**.50**	-.05	.30
MAC	.11	.03	.11	**.43**	-.12	.22
MOBL_PRG	-.03	-.01	.09	.42	.42	.36
UML	.22	.06	.20	-.02	**.59**	.44
TEAM_WK	-.00	.28	.01	.07	**.49**	.33
DB_PROG	.18	.34	.25	.09	**.41**	.39
VSL_BSC	.04	.08	-.08	.15	**.40**	.20
COBL	-.01	.10	-.08	.15	-.32	.14
Eigenvalues	8.02	2.70	2.31	2.00	1.70	
% of Variance	23.58	7.94	6.78	5.85	5.03	

Design Readiness Assessment Scale (DRAS)

Factors of the DRAS were analyzed using principal components analysis and Kaiser's normalized varimax orthogonal rotation. Employment of Kaiser's criterion of factor acceptability (associated eigenvalue greater than one) yielded five factors. Solutions of lower dimensionality and scree plots were examined and a three-factor solution yielded two factors that were uninterruptible. Loadings on these three factors are shown in Table 5.

Reliability analysis of the DRAS yielded results inconsistent with findings obtained from pilot studies. The Cronbach's alpha coefficient for the design readiness scale ($\alpha = .54$) was considerably lower than previously reported ($\alpha = .82$). Kehoe (2000) reported that test reliability values as low as .50 could be considered satisfactory for short tests (10 - 15 items). The design readiness scale fell within this range, since it contained 12 items. Kehoe also stated that important decisions concerning individual students should not be based on a single test score when the reliability rating was less than .80. It was determined that the factors of the DRAS resulted in indistinguishable subscales with low internal consistencies ($\alpha < .40$).

Although the three factors were indistinguishable when examined separately, when combined, the three factors accounted for thirty-seven percent of the variance in the DRAS. In the present study, the researcher decided against factoring the design readiness scale and proceeded with the use of the entire scale, according to the test reliability requirements cited by Kehoe.

Table 5.

Principle Components Extraction, Varimax Rotation with Kaiser Normalization – Factor Analysis of the Design Readiness Scale

		Component			Communalities
		1	2	3	
Info Hiding	DR_11	.58	-.21	.21	.42
Abstraction	DR_1	.54	.24	.05	.35
Abstraction	DR_4	.53	-.14	-.08	.30
Inhert&Plympsm	DR_5	.51	.13	-.14	.30
Encapsulation	DR_12	.42	-.02	.12	.19
Div&Conquer	DR_9	-.14	.63	.21	.46
Div&Conquer	DR_3	-.05	.62	.06	.34
Generality	DR_10	.46	.48	-.14	.46
Generality	DR_7	.36	.47	-.05	.36
Encapsulation	DR_2	.17	-.23	.76	.65
Inhert&Plympsm	DR_6	-.15	.19	.49	.30
Info Hiding	DR_8	.10	.29	.43	.39
Eigenvalues		1.92	1.40	1.13	
% of Variance		16.01	11.69	9.42	

Data analyses were conducted using all measures of cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), as well as the DRAS. Because of the low reliabilities of the logical reasoning test, prior computer science experience (various CS subscale), and the design readiness assessment scale, no claims of their predictive capabilities were considered in this research.

This study used a sample drawn from the CS2 course at two southeastern public universities. To determine the degree to which the two schools differed on the measures of cognitive abilities, prior computer science experience, and design readiness, the researcher performed a series of independent-sample *t* test. Interpretations of the means are discussed in the following sections.

Mean Analyses

Independent-sample *t* tests were conducted on measures of cognitive abilities, prior computer science experience, and design readiness.

Cognitive Abilities

Mean comparisons were conducted to determine if the school significantly differed on measures of cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style). A summary of the results is found in Table 6.

Table 6.

Mean Analysis of Cognitive Abilities, Reported by School

COGNITIVE ABILITY	SAMPLE	*N*	*M*	*SD*	*t*	*df*
Spatial Orientation						
	A	76	13.34	8.52	5.58**	159
	B	85	20.80	8.43		
Visualization						
	A	76	26.14	18.68	5.17**	159
	B	85	40.35	16.22		
Logical Reasoning						
	A	76	8.11	5.68	3.68**	159
	B	85	11.25	5.14		
Flexibility						
	A	76	18.87	5.33	4.41**	159
	B	85	22.79	5.88		
Perceptual Style						
	A	76	10.13	4.76	4.27**	159
	B	85	13.16	4.27		

**p<.01

It was found that the schools significantly differed on all measures of cognitive abilities. Participants from School A (M = 13.34, SD = 8.52) scored significantly lower on the spatial orientation test than those from School B (M = 20.80, SD = 8.43), $t(159) = 5.58$, $p = .00$. And, participants from School A (M = 26.14, SD = 18.68) scored significantly lower on the visualization test than those from School B (M = 40.35, SD = 16.22), $t(159) = 5.12$, $p = .00$. Further, participants from School A (M = 8.11, SD = 5.68), scored significantly lower on the logical reasoning test than those from School B (M = 11.25, SD = 5.41), $t(159) = 3.68$, $p = .00$. Also, participants from School A (M = 18.87, SD = 5.33) scored lower on the flexibility test than those from School B (M = 22.79, SD = 5.88), $t(159) = 4.41$, $p = .00$. Finally, participants from School A (M = 10.13, SD = 4.76) scored significantly lower on the perceptual style test than those from School B (M = 13.16, SD = 4.27), $t(159) = 4.27$, $p = .00$.

Prior Computer Science Experience

Means comparisons were conducted for each subscale of prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS). These results were coded using a Likert-type scale with values ranging from 1 (Novice) to 5 (Expert). A summary of the results is found in Table 7.

Table 7.

Mean Analysis of Prior Computer Science Experience, Reported by School

PRIOR COMPUTER SCIENCE EXPERIENCE	SAMPLE	*N*	*M*	*SD*	*t*	*df*
UNIX Programming						
	A	76	1.34	.47	2.76**	159
	B	85	1.17	.30		
Object-Oriented Processing						
	A	76	2.29	.60	4.37*	159
	B	85	2.75	.72		
Web Designing						
	A	76	2.17	.72	2.18**	159
	B	85	2.46	.85		
Computing Platform						
	A	76	2.85	.66	2.74**	159
	B	85	3.12	.61		
Various CS						
	A	76	2.69	1.34	2.76**	159
	B	85	2.56	.55		

* p < .05; **p<.01

It was found that the schools significantly differed on all subscales of prior computer science experience. Participants from School A (*M* = 1.33, *SD* = .46) had significantly more UNIX programming experience than those from School B (*M* = 1.17, *SD* = .30), $t(159) = 2.76$, $p = .01$. Further, participants from School A (*M* = 2.29, *SD* = .60) had significantly less experience in object-oriented processing than those from School B (*M* = 2.75, *SD* = .72), $t(159) = 4.37$, $p = .00$. Additionally, participants from School A (*M* = 2.17, *SD* = .85) had significantly less experience in web designing tasks than those from School B (*M* = 2.17, *SD* = .85), $t(159) = 2.18$, $p = .03$. Also, participants from School A (*M* = 2.85, *SD* = .66) had significantly less experience in computing platforms than those from School B (*M* = 3.1, *SD* = .61), $t(159) = 2.74$, $p = .01$. Finally, participants from School A (*M* = 2.69, *SD* = 1.34) had significantly less experience in various CS tasks than those from School B (*M* = 2.26, *SD* = .55), $t(159) = 2.76$, $p = .01$.

Design Readiness Assessment Scale

Means comparison was conducted to evaluate whether the two schools differed on results of the DRAS. A summary of the outcomes is found in Table 8. It was found that participants from School A (*M* = 7.09, *SD* = 1.90) scored significantly lower on the DRAS than those from School B (*M* = 8.70, *SD* = 1.90), $t(159) = 5.38$, $p = .00$.

Table 8.

Mean Analysis The Design Readiness Assessment Scale, Reported by School

DESIGN READINESS	SAMPLE	*N*	*M*	*SD*	*t*	*df*
Design Readiness Assessment Scale						
	A	76	7.09	1.90	5.38**	159
	B	85	8.70	1.90		
Pre-Training Design Task						
	A	40	11.90	3.27	3.56**	81
	B	43	14.88	4.22		
Post-Training Design Task						
	A	40	14.03	2.88	.03	81
	B	43	14.00	3.69		
Design Task Change Score						
	A	40	2.11	3.18	3.55**	81
	B	43	-.88	4.36		

* $p < .05$; ** $p < .01$

Design Task Performance

Means comparison was conducted to evaluate whether the two schools differed on results of the pre-training, post-training, and design task change scores. A summary of the outcomes is found in Table 8. It was found that the participants from School A (*M* = 11.90, *SD* = 3.27) scored significantly lower on the pre-training design task than did participants from School B (*M* =

14.00, *SD* = 3.69). However, the schools did not significantly differ on the scores of the post-training design task. Further analysis of the data found that the change from pre-training to post-training design task was significantly greater for School A (*M* = 2.11, *SD* = 3.18) than for School B (*M* = -.88, *SD* = 4.36).

To summarize, the data analyses comparing the two schools showed that there were significant differences on all measures of cognitive ability, prior computer science experience, and design readiness.

Because of the pervasive differences in the two populations, the remaining data analyses will be reported by sample (i.e., School A, School B, and Combined) according to the following research questions:

(1) What student characteristics are related to design readiness?

(2) What student characteristics are related to performance in an OOD course?

(3) What student characteristics are related to performance on a design task?

(4) Is design readiness related to a student's performance in an OOD course?

(5) Is design readiness related to a student's performance on a design task?

Research Question One: Student Characteristics Related to Design Readiness

The first area of inquiry examined the relationships among measures of student characteristics (prior computer science experience and cognitive abilities) and the design readiness assessment scale. To determine the relationship between these variables, the researcher performed a series of bivariate correlations and multiple regression analyses. The multiple regression results are presented in terms of the unstandardized beta coefficient (*B*), unstandardized coefficient standard error (*SE B*), and beta coefficient (β). The results are presented by school: School A, School B, and Combined Sample.

School A

College grade point average, cognitive ability, prior computer science experience, and the design readiness assessment scale were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and alpha coefficients of the instruments are displayed in Table 9. A review of the correlation matrix shows three variables that are significantly correlated with the design readiness assessment scale: visualization ($r(76) = .35$; $p < .01$), logical reasoning ($r(76) = .27$; $p < .05$), and object-oriented processes ($r(76) = .23$, $p < .05$). Inferences can be drawn from this data: (1) students who scored higher on the cognitive abilities measures – visualization or logical reasoning – cognitive tests were more likely to score higher on the design readiness assessment scale, and (2) students who reported higher levels of prior object-oriented processes experience were more likely to score higher on the design readiness assessment scale.

A series of multiple regression analyses were conducted to examine the relationships between the design readiness assessment scale and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), and cognitive ability (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style). The results were consistent across various set orderings. The regression equation was significant, $R^2 = .31$, adjusted $R^2 = .19$, $F(11, 64) = 2.57$, $p < .01$. The multiple regression results suggest that for School A, college grade point average, object-oriented processing, web designing, and visualization were related to the design assessment scale readiness score and accounted for 19% of the variance in the score. A summary of the results can be found in Table 10.

Table 9.

School A: Intercorrelations among College Grade Point Average, Prior Computer Science Experience, Academics, Cognitive Abilites, Design Readiness, and Course Grade.

(n = 76)

Measure		α	*M*	*SD*	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.	COLLEGE_GPA	-	2.93	.38	.05	.14	.11	.02	-.01	.03	.24*	-.12	.10	.19	.03	.42**	.20	.28*	.37**
2.	SPATIAL_ORNTN	.92	13.34	8.52	1	.55**	.31**	.27*	.33**	-.02	.10	.02	.19	.05	.18	.30**	.17	.22	.24*
3.	VISUALIZATION	.94	26.14	18.68		1	.47**	.37**	.58**	.09	.28*	.08	.12	.18	.35**	.39**	.08	.17	.44**
4.	LOGICAL_RSNING	.74	8.11	5.68			1	.29*	.22	-.08	.26*	-.06	-.01	.20	.27*	.15	-.03	-.03	.26*
5.	FLEXIBILITY	.86	18.87	5.33				1	.20	-.01	.21	.09	.2*	.24*	.14	.10	.06	-.04	.16
6.	PERCEPTL_STYLE	.83	10.13	4.76					1	.04	.19	.05	.12	-.07	.21	.21	.10	.07	.24*
7.	UNIX_PROGMMG	.76	1.34	.47						1	.40**	.49**	.44**	.18	-.08	.12	.16	-.03	.15
8.	OO_PROCSS	.78	2.29	.60							1	.44**	.26^	.27*	.23*	.34**	.36**	.12	.32**
9.	WEB_DESGNG	.84	2.17	.72								1	.45**	.24*	-.22	.01	.16	-.02	-.03
10.	COMP_PLTFRM	.71	2.85	.72									1	.31**	-.07	-.01	.07	-.12	.06
11.	VARIOUS_CS	.27	2.69	.66										1	.01	.04	-.09	.02	.08
12.	DESIGN_RDINESS	.36	7.09	1.90											1	.29*	.17	.24*	.22
13.	OVRALL_ GRADE	-	82.10	9.08												1	.52**	.74**	.81**
14.	LAB_GRADE	-	88.55	20.31													1	.39**	.16
15.	PROG_GRADE	-	83.20	14.64														1	.25*
16.	EXAM_GRADE	-	68.80	9.87															1

$^{*}p < .05$; $^{**}p < .01$

Table 10.

Summary of School A: Regression of Variables Analysis of Interest in Relation to Design Readiness.[1]

Variable	*B*	*SE B*	β
College GPA	**-.21**	**.11**	**-.23***
Prior CS Exp: UNIX Programming	-.28	.54	-.068
Prior CS Exp: OO Processing	**1.31**	**.44**	**.41****
Prior CS Exp: Web Designing	**-.97**	**.31**	**-.44****
Prior CS Exp: Computing Platform	.172	.385	.060
Prior CS Exp: Various CS	-.04	.17	-.028
Cognitive Ability: Spatial Orientation	-.07	.03	-.03
Cognitive Ability: Visualization	**.03**	**.02**	**.33***
Cognitive Ability: Logical Reasoning	.01	.04	.03
Cognitive Ability: Flexibility	-.01	.04	-.03
Cognitive Ability: Perceptual Style	-.01	.05	-.03

Notes:
[1] $R^2 = .31$; adjusted $R^2 = .19$; model significant ($p < .01$)
*($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

School B

College grade point average, cognitive abilities, prior computer science experience, and the design readiness assessment scale were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas of the instruments are displayed in Table 11. A review of the correlation matrix revealed three variables that were significantly correlated with the design readiness assessment scale: spatial orientation ($r(85) = .23$; $p < .05$), visualization ($r(85) = .27$; $p < .05$), and flexibility ($r(76) = .22$, $p < .05$). The data infers that students who scored higher on the spatial orientation, visualization, or flexibility cognitive tests were more likely to score higher on the design readiness assessment scale.

A series of multiple regression analyses were conducted to examine the relationships between the design readiness assessment scale and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), and cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style). The full regression equation was shown to be insignificant. The results were consistent across various set orderings, and a summary of the results can be found in Table 12.

Table 11.

School B: Intercorrelations among College Grade Point Average, Prior Computer Science Experience, Academics, Cognitive Abilites, Design Readiness, and Course Grade.

(n = 85)

Measure		α	*M*	*SD*	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.	COLLEGE_GPA	-	2.95	.36	.03	-.13	.06	-.07	-.09	.00	.14	-.20	-.09	-.08	.01	.50**	.19	.40**	.42**
2.	SPATIAL_ORNTN	.92	20.80	8.43	1	.58**	-.04	.22*	.38**	.20	.07	.20	-.05	.11	.23*	-.08	.05	-.06	-.03
3.	VISUALIZATION	.92	40.35	16.22		1	.17	.18	.41**	.04	-.11	.13	.03	-.02	.26*	-.04	-.01	-.08	.04
4.	LOGICAL_RSNING	.61	11.25	5.14			1	.15	-.06	.06	-.05	.03	.21	.00	.12	-.08	-.07	-.24*	.11
5.	FLEXIBILITY	.86	22.79	5.88				1	.00	.01	.07	.28*	.28**	.17	.22*	-.14	.02	-.19	-.08
6.	PERCEPTL_STYLE	.85	13.16	4.27					1	.09	.16	.18	.09	.07	.13	-.12	-.04	-.08	-.01
7.	UNIX_PROGMMG	.76	1.17	.30						1	.24*	.39*	.32**	.28*	.01	.03	-.10	.06	.11
8.	OO_PROCSS	.78	2.75	.72							1	.25*	.31**	.52**	-.03	.00	-.19	.11	.05
9.	WEB_DESGNG	.84	2.46	.85								1	.41**	.44**	-.01	-.05	-.01	.02	-.04
10.	COMP_PLTFRM	.71	3.12	.61									1	.32**	.08	-.02	-.14	.05	.01
11.	VARIOUS_CS	.27	2.56	.55										1	-.05	-.10	-.15	-.02	-.09
12.	DESIGN_RDINESS	.60	8.70	1.90											1	.11	.19	-.03	.15
13.	OVRALL_ GRADE	-	83.09	7.11												1	.34**	.78**	.81**
14.	LAB_GRADE	-	93.93	12.44													1	.01	.08
15.	PROG_GRADE	-	89.63	9.43														1	.46**
16.	EXAM_GRADE	-	76.93	7.94															1

*$p < .05$; **$p < .01$

Table 12.

Summary of School B: Regression Analysis of Variables of Interest in Relation to Design Readiness.[1]

Variable	*B*	*SE B*	β
College GPA	.02	.12	.02
Prior CS Exp: UNIX Programming	.03	.03	.13
Prior CS Exp: OO Processing	.02	.02	.13
Prior CS Exp: Web Designing	.02	.04	.06
Prior CS Exp: Computing Platform	.06	.04	.18
Prior CS Exp: Various CS	.03	.06	.06
Cognitive Ability: Spatial Orientation	.07	.80	.01
Cognitive Ability: Visualization	-.03	.36	-.01
Cognitive Ability: Logical Reasoning	-.27	.31	-.12
Cognitive Ability: Flexibility	.27	.42	.09
Cognitive Ability: Perceptual Style	-.21	.48	-.06

Notes:
[1] $R^2 = .13$; adjusted $R^2 = -.01$; model insignificant ($p = .49$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Combined Sample

College grade point average, cognitive abilities, prior computer science experience, and the design readiness assessment scale were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas the instruments are displayed in Table 13. A review of the correlation matrix demonstrated that all measures of cognitive abilities, as well as object-oriented processing, were significantly correlated with the design readiness assessment scale: spatial orientation ($r(161) = .33$; $p < .01$), visualization ($r(161) = .41$; $p < .01$), logical reasoning ($r(161) = .28$; $p < .01$), perceptual style ($r(161) = .29$; $p < .01$), and object-oriented processes ($r(161) = .20$, $p < .05$). From this data emerged two significant conclusions: (1) those that scored higher on the spatial orientation, visualization, logical reasoning, flexibility, or perceptual style cognitive tests were more likely to score higher on the design readiness assessment scale, and (2) those that reported higher levels of prior object-oriented processes experience were also more likely to score higher on the design readiness assessment scale.

A series of multiple regression analyses were conducted to examine the relationships between the design readiness assessment scale and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and school. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .28$, adjusted $R^2 = .23$, $F(12, 148) = 4.87$, $p < .01$. The multiple regression results suggest that for the Combined Sample, school and visualization had a positive and significant relationship with the design assessment scale readiness score and web-design had a negative and significant relationship with the design readiness assessment scale. These variables accounted for twenty-three percent of the variance in the score. A summary of the results can be found in Table 14.

Table 13.

Combined Sample: Intercorrelations among College Grade Point Average, Prior Computer Science Experience, Academics, Cognitive Abilites, Design Readiness, and Course Grade.

(n = 161)

Measure		α	*M*	*SD*	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.	COLLEGE_GPA	-	2.83	.47	.09	.06	.13	.02	.00	-.01	.22**	-.13	.04	.07	.05	.46**	.21**	.34**	.41**
2.	SPATIAL_ORNTN	.92	17.28	9.24	1	.63**	.23**	.34**	.44**	-.03	.21**	.18*	.15	-.03	.33**	.14	.17*	.19*	.26**
3.	VISUALIZATION	.94	33.65	18.77		1	.40**	.36**	.56**	-.02	.19*	.157*	.15	.03	.41**	.22**	.10	.16*	.38**
4.	LOGICAL_RSNING	.69	9.76	5.61			1	.29**	.17*	-.08	.18*	.04	.15	.06	.28**	.06	.00	-.03	.29**
5.	FLEXIBILITY	.85	20.95	5.95				1	.19*	-.07	.22**	.24**	.31**	.12	.29**	.00	.09	-.01	.18*
6.	PERCEPTL_STYLE	.84	11.73	4.74					1	-.01	.26**	.16*	.16*	-.1-	.28**	.08	.10	.09	.24**
7.	UNIX_PROGMMG	.76	1.24	.40						1	.22**	.38**	.32**	.24**	-.12	.07	.04	-.06	.03
8.	OO_PROCSS	.78	2.54	.70							1	.36**	.33**	.22**	.20*	.17*	.15	.18*	.29**
9.	WEB_DESGNG	.84	2.32	.86								1	.45**	.23**	-.03	-.01	.12	.04	.04
10.	COMP_PLTFRM	.76	2.99	.65									1	.23**	.09	.00	.02	.00	.12
11.	VARIOUS_CS	.27	2.46	1.03										1	-.09	-.01	-.13	-.04	-.06
12.	DESIGN_RDINESS	.54	7.94	2.06											1	.21**	.22**	.21**	.32**
13.	OVRALL_ GRADE	-	82.62	12.55												1	.46**	.74**	.76**
14.	LAB_GRADE	-	91.39	16.79													1	.31**	.18*
15.	PROG_GRADE	-	86.59	12.55														1	.39**
16.	EXAM_GRADE	-	73.09	9.76															1

**p* < .05; ***p* < .01

Table 14.

Summary of the Combined Sample: Regression Analysis of Variables of Interest in relation to Design Readiness.[1]

Variable	*B*	*SE B*	β
School	**.85**	**.38**	**.21***
College GPA	-.07	.08	-.06
Prior CS Exp: UNIX Programming	.01	.44	.00
Prior CS Exp: OO Processing	.37	.25	.13
Prior CS Exp: Web Designing	**-.47**	**.21**	**-.20***
Prior CS Exp: Computing Platform	.04	.27	.01
Prior CS Exp: Various CS	-.10	.16	-.05
Cognitive Ability: Spatial Orientation	.01	.02	.05
Cognitive Ability: Visualization	**.02**	**.01**	**.22***
Cognitive Ability: Logical Reasoning	.03	.03	.08
Cognitive Ability: Flexibility	.04	.03	.12
Cognitive Ability: Perceptual Style	.01	.04	.02

Notes:
[1] R^2 = .28; adjusted R^2 = .23; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Research Question Two: Student Characteristics Related to OOD Course Performance

The second area of inquiry examined the relationships among several student characteristics (prior computer science experience and cognitive abilities), the design readiness assessment scale, and OOD course performance. OOD course performance is represented by four variables: lab grade, project grade, exam grade, and overall grade. To determine the relationship between these variables, the researcher performed a series of bivariate correlation and multiple regression analyses. The results are presented by school: School A, School B, and Combined Sample. Within each school, the data is also presented according to each measure of OOD course performance.

School A

College grade point average, cognitive abilities, prior computer science experience, the design readiness assessment scale, lab grades, project grades, exam grade, and overall grade comprised the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas of the instruments are displayed in Table 9.

Lab Grade

A review of the correlation matrix showed two variables that were significantly correlated with the course lab grade: object-oriented processes ($r(76) = .36$, $p < .01$) and overall grade ($r(76) = .52$, $p < .01$). From this data, one can infer that (1) students who reported higher levels of prior object-oriented processes experience were more likely to score higher on the course labs, and (2) students who scored higher in their overall course grade were more likely to score higher on the course labs.

A series of multiple regression analyses was conducted to examine the relationships between the lab grades and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .27$, adjusted $R^2 = .13$, $F(12, 63) = 1.91$, $p < .05$. The multiple regression results suggested that for School A, object-oriented processing had a positive and significant association with lab grades, and accounted for thirteen percent of the variance in the grade. A summary of the results can be found in Table 15.

Table 15.

Summary of the School A: Regression Analysis of Variables of Interest in relation OOD Performance – Lab Grade.[1]

Variable	*B*	*SE B*	β
College GPA	2.25	1.23	.22
Prior CS Exp: UNIX Programming	2.43	5.99	.06
Prior CS Exp: OO Processing	**10.11**	**5.15**	**.30***
Prior CS Exp: Web Designing	3.70	3.69	.16
Prior CS Exp: Computing Platform	-3.10	4.28	-.10
Prior CS Exp: Various CS	-2.94	1.88	-.19
Cognitive Ability: Spatial Orientation	.56	.32	-.24
Cognitive Ability: Visualization	-.19	.19	-.18
Cognitive Ability: Logical Reasoning	-.55	.47	-.15
Cognitive Ability: Flexibility	.26	.47	.07
Cognitive Ability: Perceptual Style	.17	.59	.04
Design Readiness Assessment Scale	1.93	1.39	.18

Notes:
[1] $R^2 = .27$; adjusted $R^2 = .13$; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Project Grade

A review of the correlation matrix showed four variables that were found to be significantly correlated with the course project grade: college grade point average ($r(76) = .28$, *p*

< .05), design readiness assessment scale ($r(76) = .24$, $p < .05$), lab grade ($r(76) = .40$, $p < .01$), and overall grade ($r(76) = .74$, $p < .01$). From this data one can infer that (1) students with a higher college grade point average were more likely to score higher on the course projects, (2) students that scored higher on the design readiness assessment scale were more likely to score higher on the course projects, and (3) not surprisingly, those that scored higher on the course labs and on the overall course grade were more likely to score higher on the course projects.

A series of multiple regression analyses were conducted to examine the relationship between the project grades and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .26$, adjusted $R^2 = .12$, $F(12, 63) = 1.88$, $p < .05$. The multiple regression results suggested that for School A, college grade point average and the design readiness assessment scale had positive and significant relationship with project grades, and accounted for twelve percent of the variance in the grade. A summary of the results can be found in Table 16.

Table 16.

Summary of the School A: Regression Analysis of Variables of Interest in relation OOD Performance – Project Grade.[1]

Variable	*B*	*SE B*	β
College GPA	**2.52**	**.89**	**.35****
Prior CS Exp: UNIX Programming	-.27	4.32	-.01
Prior CS Exp: OO Processing	-.622	3.72	-.025
Prior CS Exp: Web Designing	3.48	2.66	.20
Prior CS Exp: Computing Platform	-5.92	3.09	-.27
Prior CS Exp: Various CS	.39	1.36	.04
Cognitive Ability: Spatial Orientation	.01	.14	.02
Cognitive Ability: Visualization	.012	.139	.02
Cognitive Ability: Logical Reasoning	-.55	.34	-.21
Cognitive Ability: Flexibility	-.15	.34	-.05
Cognitive Ability: Perceptual Style	-.01	.43	.00
Design Readiness Assessment Scale	**2.27**	**1.00**	**.29***

Notes:
[1] $R^2 = .26$; adjusted $R^2 = .12$; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Exam Grade

A review of the correlation matrix shows eight variables that were significantly correlated with the course exam grade: college grade point average ($r(76) = .37$, $p < .01$), visualization

($r(76) = .24$, $p < .05$), surface development ($r(76) = .44$, $p < .01$), logical reasoning ($r(76) = .26$, $p < .05$), perceptual style ($r(76) = .24$, $p < .05$), object-oriented processing ($r(76) = .32$, $p < .01$), project grade ($r(76) = .25$, $p < .05$), and overall grade ($r(76) = .81$, $p < .01$). From this data, one can infer that (1) those with a higher college grade point average were more likely to score higher on the course exams, (2) those that scored higher on the cognitive measures of spatial orientation, visualization, logical reasoning, or perceptual style were more likely to score higher on the course exams, (3) those the reported higher levels of object-oriented processing experience were more likely to score higher on the course exams, and (4) those that scored higher on project grades were more likely to score higher on the course exams.

A series of multiple regression analyses were conducted to examine the relationship between the exam grades and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .34$, adjusted $R^2 = .22$, $F(12, 63) = 2.75$, $p < .01$. The multiple regression results suggested that for School A, college grade point average had a positive and significant association with exam grades, and accounted for twenty-two percent of the variance in the grade. A summary of the results can be found in Table 17.

Table 17.

Summary of the School A: Regression Analysis of Variables of Interest in relation OOD Performance – Exam Grade.[1]

Variable	*B*	*SE B*	β
College GPA	**1.40**	**.57**	**.29****
Prior CS Exp: UNIX Programming	3.13	2.75	.15
Prior CS Exp: OO Processing	2.55	2.37	.15
Prior CS Exp: Web Designing	-1.28	1.70	-.11
Prior CS Exp: Computing Platform	-.798	1.97	-.05
Prior CS Exp: Various CS	-.54	.86	-.07
Cognitive Ability: Spatial Orientation	.04	.15	.03
Cognitive Ability: Visualization	.17	.09	.32
Cognitive Ability: Logical Reasoning	.06	.21	.03
Cognitive Ability: Flexibility	.05	.22	.02
Cognitive Ability: Perceptual Style	.05	.22	-.01
Design Readiness Assessment Scale	.24	.64	.05

Notes:
[1] $R^2 = .34$; adjusted $R^2 = .22$; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Overall Grade

A review of the correlation matrix revealed four variables that were significantly correlated with the overall course grade: college grade point average ($r(76) = .42$, $p < .01$), spatial orientation ($r(76) = .30$, $p < .01$), visualization ($r(76) = .39$, $p < .01$), object-oriented processes ($r(76) = .34$, $p < .01$), and the design readiness assessment scale ($r(76) = .29$, $p < .01$). From this data, one can infer that (1) those with a higher college grade point average were more likely to score higher on their overall course grade, (2) those that scored higher on the cognitive measures of spatial orientation or visualization were more likely to score higher on their overall course grade, (3) those that reported higher levels of object-oriented processing experience were more likely to score higher on their overall course grade, and (4) those that scored higher on the design readiness assessment scale were more likely to score higher on their overall course grade.

A series of multiple regression analyses were conducted to examine the relationship between overall course grade, college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .41$, adjusted $R^2 = .30$, $F(12, 63) = 3.69$, $p < .01$. The multiple regression results suggested that only college grade point average had a positive and significant association with the overall grade in the course, and accounted for thirty percent of the variance in a students' final course grade. A summary of the results can be found in Table 18.

Table 18.

Summary of the School A: Regression Analysis of Variables of Interest in relation OOD Performance – Overall Grade.[1]

Variable	*B*	*SE B*	β
College GPA	**1.82**	**.49**	**.41****
Prior CS Exp: UNIX Programming	2.04	2.39	.11
Prior CS Exp: OO Processing	2.35	2.06	.15
Prior CS Exp: Web Designing	.64	1.46	.06
Prior CS Exp: Computing Platform	-2.57	1.71	-.19
Prior CS Exp: Various CS	-.50	.75	-.07
Cognitive Ability: Spatial Orientation	.22	.13	.21
Cognitive Ability: Visualization	.09	.08	.18
Cognitive Ability: Logical Reasoning	-.19	.19	-.12
Cognitive Ability: Flexibility	.01	.19	.01
Cognitive Ability: Perceptual Style	.00	.24	.00
Design Readiness Assessment Scale	1.01	.55	.21

Notes:
[1] $R^2 = .41$; adjusted $R^2 = .30$; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

School B

College grade point average, cognitive abilities, prior computer science experience, the design readiness assessment scale, lab grade, project grade, exam grade, and overall grade were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas of the instruments are displayed in Table 11.

Lab Grade

A review of the correlation matrix revealed only one variable that significantly correlated with the course lab grade: overall grade ($r(85) = .34$, $p < .01$). This relation suggests that students with a higher college grade point average were more likely to score higher on the course labs.

A series of multiple regression analyses were conducted to examine the relationship between the lab grades and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was insignificant. A summary of the results can be found in Table 19.

Table 19.

Summary of the School B: Regression Analysis of Variables of Interest in relation OOD Performance – Lab Grade.[1]

Variable	*B*	*SE B*	β
College GPA	1.56	.76	.24
Prior CS Exp: UNIX Programming	-3.46	5.18	-.08
Prior CS Exp: OO Processing	-3.48	2.35	-.20
Prior CS Exp: Web Designing	2.78	2.03	.19
Prior CS Exp: Computing Platform	-1.86	2.74	-.09
Prior CS Exp: Various CS	-1.18	3.13	-.05
Cognitive Ability: Spatial Orientation	.04	.22	.03
Cognitive Ability: Visualization	-.05	.11	-.07
Cognitive Ability: Logical Reasoning	-.21	.28	-.09
Cognitive Ability: Flexibility	.02	.26	.01
Cognitive Ability: Perceptual Style	-.06	.37	-.02
Design Readiness Assessment Scale	1.36	.76	.21

Notes:
[1] $R^2 = .16$; adjusted $R^2 = .04$; model not significant ($p = .34$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Project Grade

A review of the correlation matrix showed three variable that were significantly correlated with the course project grade: college grade point average ($r(85) = .40$, $p < .01$), logical reasoning ($r(85) = -.25$, $p < .05$), and overall grade ($r(85) = .78$, $p < .01$). Interestingly, that data inferred that (1) those who scored *lower* on the cognitive abilities measure of logical reasoning were more likely to score higher on the course projects, (2) those with a higher college grade point average were more likely to score higher on the course projects, and (3) not surprisingly, those who scored higher on the overall course grade were more likely to score higher on the course projects.

A series of multiple regression analyses were conducted to examine the relationship between the project grades and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .31$, adjusted $R^2 = .20$, $F(12, 72) = 2.70$, $p < .01$. The multiple regression results suggested that for School B, college grade point average had a positive and significant relationship, while logical reasoning had a negative significant relationship with project grades and accounted for twenty percent of the variance in the project grade. A summary of the results can be found in Table 20.

Table 20.

Summary of the School B: Regression Analysis of Variables of Interest in relation OOD Performance – Project Grade.[1]

Variable	*B*	*SE B*	β
College GPA	**2.23**	**.52**	**.45****
Prior CS Exp: UNIX Programming	.07	3.56	.00
Prior CS Exp: OO Processing	.49	1.61	.04
Prior CS Exp: Web Designing	1.85	1.40	.17
Prior CS Exp: Computing Platform	2.20	1.89	.14
Prior CS Exp: Various CS	-1.08	2.15	-.06
Cognitive Ability: Spatial Orientation	-.13	.15	-.11
Cognitive Ability: Visualization	.09	.09	.15
Cognitive Ability: Logical Reasoning	**-.58**	**.19**	**-.31****
Cognitive Ability: Flexibility	-.32	.18	-.20
Cognitive Ability: Perceptual Style	-.27	.25	-.12
Design Readiness Assessment Scale	.21	.52	.04

Notes:
[1] $R^2 = .31$; adjusted $R^2 = .20$; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Exam Grade

A review of the correlation matrix showed only one variable that was significantly correlated with the exam grade: college grade point average ($r(85) = .42$, $p < .01$). This relation suggests that students with a higher college grade point average were more likely to score higher on course exams.

A series of multiple regression analyses were conducted to examine the relationship between exam grade and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .25$, adjusted $R^2 = .13$, $F(12, 63) = 2.02$, $p < .05$. The multiple regression results suggested that for School B, college grade point average had a positive and significant relationship with exam grades, and accounted for thirteen percent of the variance in the grade. A summary of the results can be found in Table 21.

Table 21.

Summary of the School B: Regression Analysis of Variables of Interest in relation OOD Performance – Exam Grade.[1]

Variable	*B*	*SE B*	β
College GPA	**1.82**	**.46**	**.44***
Prior CS Exp: UNIX Programming	3.39	3.12	.13
Prior CS Exp: OO Processing	.43	1.41	.04
Prior CS Exp: Web Designing	.80	1.23	.09
Prior CS Exp: Computing Platform	-.21	1.65	-.02
Prior CS Exp: Various CS	-1.37	1.89	-.10
Cognitive Ability: Spatial Orientation	-.170	.13	-.18
Cognitive Ability: Visualization	.079	.07	.16
Cognitive Ability: Logical Reasoning	.056	.17	.04
Cognitive Ability: Flexibility	-.11	.16	-.08
Cognitive Ability: Perceptual Style	-.04	.22	-.02
Design Readiness Assessment Scale	.672	.46	.16

Notes:
[1] R^2 = .25; adjusted R^2 = .13; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Overall Grade

A review of the correlation matrix showed only one variable that was significantly correlated with the overall course grade: college grade point average ($r(85) = .42$, $p < .01$). This relationship suggests that students with a higher college grade point average were more likely to obtain a higher grade at the end of the course.

A series of multiple regression analyses were conducted to examine the relationship between the overall course grade and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .34$, adjusted $R^2 = .23$, $F(12, 72) = 3.11$, $p < .01$. The multiple regression results suggested that only college grade point average had a positive and significant association with the overall grade in the course, and accounted for twenty-three percent of the variance in the grade. A summary of the results can be found in Table 22.

Table 22.

Summary of the School B: Regression Analysis of Variables of Interest in relation OOD Performance – Overall Grade.[1]

Variable	*B*	*SE B*	β
College GPA	**2.04**	**.38**	**.55****
Prior CS Exp: UNIX Programming	.67	2.62	.03
Prior CS Exp: OO Processing	-.46	1.19	-.05
Prior CS Exp: Web Designing	1.28	1.03	.15
Prior CS Exp: Computing Platform	.54	1.39	.05
Prior CS Exp: Various CS	-.71	1.58	-.06
Cognitive Ability: Spatial Orientation	-.15	.11	-.18
Cognitive Ability: Visualization	.08	.06	.18
Cognitive Ability: Logical Reasoning	-.24	.14	-.17
Cognitive Ability: Flexibility	-.18	.13	-.15
Cognitive Ability: Perceptual Style	-.22	.19	-.13
Design Readiness Assessment Scale	.61	.38	.16

Notes:
[1] R^2 = .34; adjusted R^2 = .23; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Combined Sample

College grade point average, cognitive abilities, prior computer science experience, the design readiness assessment scale, lab grade, project grade, exam grade, and overall grade were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas of the instruments are displayed in Table 13.

Lab Grade

A review of the correlation matrix showed four variables that were significantly correlated with the course lab grade: college grade point average ($r(161) = .46$, $p < .01$), visualization ($r(161) = .17$, $p < .05$), the design readiness assessment scale ($r(161) = .22$, $p < .05$), and overall grade ($r(161) = .46$, $p < .01$). These relations suggest that (1) students with a higher college grade point average were more likely to score higher on the course labs, (2) students who scored higher on the cognitive measure of visualization were more likely to score higher on the course labs, (3) students who scored higher on the design readiness assessment scale were more likely to score higher on the course labs, and (4) students who scored higher on the overall course grade were more likely to score higher on the lab grade.

A series of multiple regression analyses were conducted to examine the relationship between the lab grades and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .16$, adjusted $R^2 = .08$, $F(13, 147) = 2.12$, $p < .05$. The multiple regression results suggested that for the combined sample, college grade point average, and the design readiness assessment scale had positive had significant relationship, and various CS experience had a negative and significant relationship with lab grade. This model accounted for only eight percent of the variance in the lab grades. A summary of the results can be found in Table 23.

Table 23.

Summary of the Combined Sample: Regression Analysis of Variables of Interest in relation OOD Performance – LabGrade.[1]

Variable	*B*	*SE B*	β
School	-.60	3.41	-.02
College GPA	**1.99**	**.69**	**.24***
Prior CS Exp: UNIX Programming	2.02	3.89	.05
Prior CS Exp: OO Processing	1.21	2.25	.05
Prior CS Exp: Web Designing	3.62	1.91	.19
Prior CS Exp: Computing Platform	-2.18	2.42	-.08
Prior CS Exp: Various CS	**-2.87**	**1.42**	**-.18***
Cognitive Ability: Spatial Orientation	.17	.19	.09
Cognitive Ability: Visualization	-.04	.11	-.04
Cognitive Ability: Logical Reasoning	-.24	.26	-.08
Cognitive Ability: Flexibility	.11	.25	.04
Cognitive Ability: Perceptual Style	-.04	.34	-.01
Design Readiness Assessment Scale	**1.70**	**.73**	**.21***

Notes:
[1] R^2 = .16; adjusted R^2 = .08; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Project Grade

A review of the correlation matrix revealed seven variables that were significantly correlated with the course project grade: college grade point average ($r(161) = .21$, $p < .01$), spatial orientation ($r(161) = .19$, $p < .05$), visualization ($r(161) = .16$, $p < .05$), object-oriented processing ($r(161) = .18$, $p < .01$), the design readiness assessment scale ($r(161) = .21$, $p < .01$), lab grade ($r(161) = .79$, $p < .01$), and overall grade ($r(161) = .31$, $p < .01$). These relations suggest (1) those with a higher college grade point average were more likely to score higher on the course projects, (2) students who scored higher on the cognitive abilities measure of visualization were more likely to score higher on the course projects, (3) students who scored higher on the design readiness assessment scale were more likely to score higher on the course projects, and (4), not surprisingly, students who scored higher on the overall course grade were more likely to score higher on the course projects.

A series of multiple regression analyses were conducted to examine the relationship between the project grades and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .24$, adjusted $R^2 = .17$, $F(13, 147) = 3.56$, $p < .01$. The multiple regression results suggest that for the combined sample, college grade point average and the design readiness assessment scale had positive and significant association, while logical reasoning had a negative significant association with project grades, and accounted for seventeen percent of the variance in the grade. A summary of the results can be found in Table 24.

Table 24.

Summary of the Combined Sample: Regression Analysis of Variables of Interest in relation OOD Performance – Project Grade.[1]

Variable	*B*	*SE B*	β
School	3.69	2.42	.15
College GPA	**2.10**	**.49**	**.33****
Prior CS Exp: UNIX Programming	-1.98	2.77	-.06
Prior CS Exp: OO Processing	1.19	1.60	.07
Prior CS Exp: Web Designing	1.69	1.36	.12
Prior CS Exp: Computing Platform	-.98	1.72	-.05
Prior CS Exp: Various CS	-.18	1.01	-.02
Cognitive Ability: Spatial Orientation	.09	.13	.07
Cognitive Ability: Visualization	.08	.08	.12*
Cognitive Ability: Logical Reasoning	**-.41**	**.18**	**-.18***
Cognitive Ability: Flexibility	-.29	.18	-.14
Cognitive Ability: Perceptual Style	-.18	.24	-.07
Design Readiness Assessment Scale	**1.02**	**.52**	**.17***

Notes:
[1] $R^2 = .24$; adjusted $R^2 = .17$; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Exam Grade

A review of the correlation matrix showed 11 variables that were significantly correlated with the course exam grade: college grade point average ($r(161) = .46$, $p < .01$), spatial orientation ($r(161) = .26$, $p < .01$), visualization ($r(161) = .38$, $p < .01$), logical reasoning ($r(161) = .29$, $p < .01$), flexibility ($r(161) = .18$, $p < .01$), perceptual style ($r(161) = .24$, $p < .01$), object-oriented processing ($r(161) = .29$, $p < .01$), the design readiness assessment scale ($r(161) = .32$, $p < .01$), lab grade ($r(161) = .76$, $p < .01$), project grade ($r(161) = .18$, $p < .05$), and overall course grade($r(161) = .39$, $p < .01$). These relations suggest that (1) students with a higher college grade point average were more likely to score higher on the course exams, (2) students who scored higher on the cognitive abilities measure spatial orientation, visualization, logical reasoning, flexibility, or perceptual style were more likely to score higher on the exam grade, and (3) students who reported higher level of object-oriented processing experience were more likely to score higher on the course exams, (4) students who scored higher on the design readiness assessment scale were more likely to score higher on the course exams, and (5), not surprisingly, students who scored higher on labs, projects, and overall in the course were more likely to score higher on the course exams.

A series of multiple regression analyses were conducted to examine the relationship between exam grade and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .40$, adjusted $R^2 = .34$, $F(13, 147) = 7.41$, $p < .01$. The multiple regression results suggest that for the combined sample, school, college grade point average, and the cognitive ability visualization had positive and significant association with exam grades, and accounted for thirty-four percent of the variance in the grade. A summary of the results can be found in Table 25.

Table 25.

Summary of the Combined Sample: Regression Analysis of Variables of Interest in relation OOD Performance – Exam Grade.[1]

Variable	*B*	*SE B*	β
School	**4.80**	**1.68**	**.25****
College GPA	**1.69**	**.34**	**.34****
Prior CS Exp: UNIX Programming	3.07	1.92	.12
Prior CS Exp: OO Processing	1.05	1.11	.08
Prior CS Exp: Web Designing	-.26	.94	-.02
Prior CS Exp: Computing Platform	-.41	1.19	-.03
Prior CS Exp: Various CS	-.62	.70	-.07
Cognitive Ability: Spatial Orientation	-.08	.09	-.08
Cognitive Ability: Visualization	**.13**	**.05**	**.24***
Cognitive Ability: Logical Reasoning	.13	.13	.08
Cognitive Ability: Flexibility	-.03	.13	-.02
Cognitive Ability: Perceptual Style	.01	.17	.00
Design Readiness Assessment Scale	.53	.36	.11

Notes:
[1] R^2 = .40; adjusted R^2 = .34; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Overall Grade

A review of the correlation matrix showed seven variables that were significantly correlated with the overall course grade: college grade point average ($r(161) = .41$, $p < .01$), visualization ($r(161) = .21$, $p < .01$), object-oriented processing ($r(161) = .17$, $p < .05$), the design readiness assessment scale ($r(161) = .22$, $p < .01$), lab grade ($r(161) = .46$, $p < .01$), project grade ($r(161) = .74$, $p < .01$), and exam grade ($r(161) = .76$, $p < .01$),. These relations suggest that (1) students with a higher college grade point average were more likely to receive a higher overall course grade, (2) students who scored higher on the cognitive abilities measure visualization were more likely to receive a higher overall course grade, and (3) students who reported higher levels of object-oriented processing experience were more likely to receive a higher overall course grade, (4) students who scored higher on the design readiness assessment scale were more likely to score higher on the course exams, and (5), not surprisingly, students who scored higher on labs, projects, and exams were more likely to receive a higher overall course grade.

A series of multiple regression analyses were conducted to examine the relationship between the overall course grade and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style), and the design readiness assessment scale. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .30$, adjusted $R^2 = .24$, $F(13, 147) = 7.41$, $p < .01$. The multiple regression results suggested that college grade point average, the cognitive abilities measure of visualization, and the design readiness assessment scale had positive and significant association with the overall grade in the course, and accounted for twenty-four percent of the variance in the grade. A summary of the results can be found in Table 26.

Table 26.

Summary of the Combined Sample: Regression Analysis of Variables of Interest in relation OOD Performance – Overall Grade.[1]

Variable	*B*	*SE B*	β
School	-2.15	1.50	-.13
College GPA	**1.89**	**.30**	**.46****
Prior CS Exp: UNIX Programming	1.12	1.71	.06
Prior CS Exp: OO Processing	.92	.99	.08
Prior CS Exp: Web Designing	.70	.84	.08
Prior CS Exp: Computing Platform	-.71	1.06	-.06
Prior CS Exp: Various CS	-.67	.62	-.08
Cognitive Ability: Spatial Orientation	-.02	.08	-.02
Cognitive Ability: Visualization	**.10**	**.05**	**.24***
Cognitive Ability: Logical Reasoning	-.10	.11	-.07
Cognitive Ability: Flexibility	-.10	.11	-.07
Cognitive Ability: Perceptual Style	-.11	.15	-.06
Design Readiness Assessment Scale	**.79**	**.32**	**.20****

Notes:
[1] R^2 = .30; adjusted R^2 = .24; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Research Question Three: Student Characteristics Related to Design Task

The third area of inquiry targeted the extent to which student characteristics (background and cognitive state) were related to performance on the post-training Design Task. The pre-training design task score was entered into the analyses as a control variable. To determine the relationship between these variables, the researcher performed a series of bivariate correlation and multiple regression analyses. The results are presented by school: School A, School B, and Combined Sample. Due to design task grader limitations, a convenience sample was drawn from both schools (approximately 50%). The design tasks were graded by three trained graders using the SIP grading rubric (Custer, Valesey, & Burke, 2001). The inter-rater reliability ($\alpha = .78$) of the graders was found to be well above acceptable minimum inter-rater reliability requirements (Nunnally, 1978).

School A

The relationship between performance on the design task and college grade point average, cognitive abilities, prior computer science experience, and the design readiness assessment scale were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas of the instruments are displayed in Table 27. A review of the correlation matrix showed three variables that were significantly correlated with performance on the post-training design task: visualization ($r(40) = .47$; $p < .01$), flexibility ($r(40) = .33$; $p < .05$), and pre-training design task ($r(40) = .47$, $p < .01$). These relations suggest that (1) students who scored higher on the visualization or flexibility cognitive tests were more likely to score higher on post-training design task and (2), not surprisingly, students who scored higher on pre-training design task were more likely to score higher on the post-training design task.

Table 27.

School A: Intercorrelations among Measures of Demographics, Prior Computer Science Experience, Academics, Cognitive Abilites, Personality, Design Readiness, and Design Task.

(n = 40)

Measure	α	*M*	*SD*	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1. COLLEGE_GPA	-	2.93	.38	.28	.28	.08	.03	.13	-.21	.21	-.03	.14	.22	.03	.40*	.24	-.19
2. SPATIAL_ORNTN	.92	13.34	8.52	1	.62**	.44**	.51**	.29	.08	.26	.09	.34*	.14	.12	.16	.47**	.26
3. VISUALIZATION	.94	26.14	18.68		1	.44**	.58**	.62**	.04	.32*	.13	.35*	.13	.35*	.14	.29	.13
4. LOGICAL_RSNING	.74	8.11	5.68			1	.48**	.34*	.08	.23	.04	.15	.12	.29	.03	.27	.22
5. FLEXIBILITY	.86	18.87	5.33				1	.38*	.01	.30	.13	.32*	.22	.31	.12	.33*	.18
6. PERCEPTL_STYLE	.83	10.13	4.76					1	-.05	.37*	.17	.29	.20	.21	.11	.22	.09
7. UNIX_PROGMMG	.76	1.34	.47						1	.45**	.42**	.38*	.27	.1	-.08	-.22	-.12
8. OO_PROCSS	.78	2.29	.60							1	.55**	.35*	.63**	.32*	-.14	-.03	.12
9. WEB_DESGNG	.84	2.17	.72								1	.45**	.63**	-.05	-.02	.05	.07
10. COMP_PLTFRM	.71	2.85	.72									1	.47**	.15	.24	-.02	-.27
11. VARIOUS_CS	.27	2.69	.66										1	.05	.14	.03	-.11
12. DESIGN_RDINESS	.36	7.09	1.90											1	-.12	.04	.16
13. Pre-Training Task Score	.76	-	-												1	.47**	-.60**
14. Post-Training Task Score	.83	-	-													1	.42**
15. Design Task Change																	1

**p* < .05
***p* < .01

A series of multiple regression analyses were conducted to examine the relationship between the post-training design task score and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), and cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style). The results were consistent across various set orderings. The regression equation was significant, $R^2 = .58$, adjusted $R^2 = .37$, $F(13, 26) = 2.74$, $p < .01$. The multiple regression results suggested that for School A, pre-training design task and the prior computer science experience measure of computing platforms had positive and significant relationships to post-training design task, while the cognitive ability measure of spatial orientation was negative and significantly related to the post-training design task. These measures accounted for thirty-seven percent of the variance in the post-training design task score. A summary of the results can be found in Table 28.

Table 28.

Summary of the School A: Regression Analysis of Variables of Interest in relation Design Task Performance.[1]

Variable	*B*	*SE B*	β
College GPA	.00	.28	.00
Pre-Training Task Score	**.43**	**.14**	**.48****
Prior CS Exp: UNIX Programming	.18	.06	.54
Prior CS Exp: OO Processing	-.03	.03	-.20
Prior CS Exp: Web Designing	.02	.08	.04
Prior CS Exp: Computing Platform	**.07**	**.10**	**.13****
Prior CS Exp: Various CS	.09	.11	.15
Cognitive Ability: Spatial Orientation	**-1.34**	**1.70**	**-.14****
Cognitive Ability: Visualization	-.67	1.13	-.15
Cognitive Ability: Logical Reasoning	1.35	.71	.37
Cognitive Ability: Flexibility	-1.93	.83	-.42
Cognitive Ability: Perceptual Style	-.22	.780	-.06
Design Readiness Assessment Scale	.24	.23	.17

Notes:
[1] R^2 = .58; adjusted R^2 = .37; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

School B

The relationship between post-training design task, college grade point average, cognitive abilities, prior computer science experience, and the design readiness assessment scale were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas of the instruments are displayed in Table 29. A review of the correlation matrix shows two variables that were significantly correlated with the post-training design task score: object-oriented processing ($r(43) = .30$; $p < .01$) and pre-training design task score ($r(43) = .40$; $p < .01$). These relations suggest that (1) students who reported higher levels of prior computer science experience (object- oriented processing) were more likely to score higher on the post-training design task, and (2), not surprisingly, students who scored higher on the pre-training design task were more likely to score higher on the post-training design task score.

A series of multiple regression analyses were conducted to examine the relationship between the post-training design task score and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), and cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style). The full regression equation was shown to be insignificant. Several measures were shown to be significant, however their use is meaningless. The overall model can only classify approximately ninety -two percent of the students, where the minimum statistical classification is ninety-five percent (Nunnally, 1978). The results were consistent across various regression set orderings. A summary of the results can be found in Table 30.

Table 29.

School B: Intercorrelations Among Measures of Demographics, Prior Computer Science Experience, Academics, Cognitive Abilites, Personality, Design Readiness, and Design Task.

(n = 43)

Measure		α	*M*	*SD*	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1.	COLLEGE_GPA	-	2.95	.36	.07	-.16	-.00	-.07	-.03	-.04	.27	-.21	-.09	.02	.08	-.13	.22	.31*
2.	SPATIAL_ORNTN	.92	20.80	8.43	1	.56**	-.20	.27	.43**	.22	.02	.26	-.07	.05	.20	.05	.06	.00
3.	VISUALIZATION	.92	40.35	16.22		1	.05	.30	.55**	.14	.02	.27	.18	.16	.34*	-.01	-.06	-.04
4.	LOGICAL_RSNING	.61	11.25	5.14			1	.291	-.12	.14	-.04	.23	.41**	.14	.06	.15	.06	-.10
5.	FLEXIBILITY	.86	22.79	5.88				1	.06	.03	-.05	.41**	.37*	.13	.36*	.06	.14	.06
6.	PERCEPTL_STYLE	.85	13.16	4.27					1	.04	.12	.09	.10	-.04	.15	-.01	.00	.00
7.	UNIX_PROGMMG	.76	1.17	.30						1	.31*	.40**	.32*	.46**	.03	.13	.15	.00
8.	OO_PROCSS	.78	2.75	.72							1	.130	.21	.27	-.02	-.06	.30*	.32*
9.	WEB_DESGNG	.84	2.46	.85								1	.37*	.34*	.02	.04	.04	-.00
10.	COMP_PLTFRM	.71	3.12	.61									1	.25	.11	.13	-.13	-.24
11.	VARIOUS_CS	.27	2.56	.55										1	-.09	-.18	.01	.18
12.	DESIGN_RDINESS	.60	8.70	1.90											1	.17	.08	-.10
13.	Pre-Training Task Score	.81	-	-												1	.40**	-.63**
14.	Post-Training Task Score	.83	-	-													1	.46**
15.	Design Task Change Score																	1

*$p < .05$
**$p < .01$

Table 30.

Summary of the School B: Regression Variables of Analysis of Interest in relation Design Task Performance.[1]

Variable	*B*	*SE B*	β
College GPA	.36	.29	.20**
Pre-Training Task Score	.42	.13	.48
Prior CS Exp: UNIX Programming	-.07	.09	-.15
Prior CS Exp: OO Processing	.00	.05	.00*
Prior CS Exp: Web Designing	.04	.13	.05
Prior CS Exp: Computing Platform	.21	.11	.34*
Prior CS Exp: Various CS	.05	.14	.06
Cognitive Ability: Spatial Orientation	1.88	2.34	.15
Cognitive Ability: Visualization	1.73	.80	.34
Cognitive Ability: Logical Reasoning	.09	.84	.02
Cognitive Ability: Flexibility	-2.86	1.13	-.45
Cognitive Ability: Perceptual Style	-.09	1.35	-.01
Design Readiness Assessment Scale	-.12	.28	-.07

Notes:
[1] $R^2 = .45$; adjusted $R^2 = .21$; model insignificant ($p = .08$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Combined Sample

The relationship between post-training design task, college grade point average, cognitive abilities, prior computer science experience, the design readiness assessment scale, and school were the variables of interest for this research question. The means, standard deviations, and bivariate correlations of the variables and coefficient alphas the instruments are displayed in Table 31. A review of the correlation matrix showed only two variables significantly correlated with the post-training design task score: college grade point average ($r(83) = .23$; $p < .05$) and pre-training design task score ($r(83) = .39$, $p < .01$). These relations suggest that (1) students with a higher college grade point average were more likely to score higher on the post-training design task, and (2) students who scored higher on the pre-training design task score were more likely to score higher on the post-training design task score.

A series of multiple regression analyses were conducted to examine the relationship between the post-training design task score and college grade point average, prior computer science experience (UNIX programming, object-oriented processes, web designing, computing platforms, and various CS), and cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, and perceptual style). School was also included as a control variable in these analyses. The results were consistent across various set orderings. The regression equation was significant, $R^2 = .39$, adjusted $R^2 = .26$, $F(14, 68) = 3.03$, $p < .01$. The multiple regression results suggested that for the combined sample, pre-training design task and the prior computer science of computing platforms were positive and significant in relation to post-training design task, while school and the prior computer science experience measure of object-oriented processing were negative and significantly related to the post-training design task. These measures accounted for twenty-six percent of the variance in the post-training design task score.

A summary of the results can be found in Table 32.

Table 31.

Combined Sample: Intercorrelations among Measures of Demographics, Prior Computer Science Experience, Academics, Cognitive Abilites, Personality, Design Readiness, and Design Task.

(n = 83)

Measure	α	*M*	*SD*	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1. COLLEGE_GPA	-	2.83	.47	.20	.12	.08	.015	.081	-.15	.26*	-.11	.04	.10	.09	.13	.23*	.06
2. SPATIAL_ORNTN	.92	17.28	9.24	1	.68**	.28**	.47**	.44**	.02	.25*	.21	.21	-.03	.28**	.25*	.21	-.08
3. VISUALIZATION	.94	33.65	18.77		1	.40**	.52**	.63**	-.04	.28*	.22*	.31**	-.00	.44**	.22*	.10	-.14
4. LOGICAL_RSNING	.69	9.76	5.61			1	.45**	.22*	.01	.19	.16	.30**	.02	.27*	.21	.14	-.09
5. FLEXIBILITY	.85	20.95	5.95				1	.29**	-.06	.18	.30**	.37**	.08	.41**	.19	.20	-.02
6. PERCEPTL_STYLE	.84	11.73	4.74					1	-.07	.30**	.16	.24*	.03	.25*	.14	.09	-.07
7. UNIX_PROGMMG	.76	1.24	.40						1	.28*	.37**	.30**	.38**	.01	-.05	-.01	.04
8. OO_PROCSS	.78	2.54	.70							1	.33**	.31**	.34**	.21	.03	.17	.11
9. WEB_DESGNG	.84	2.32	.86								1	.42**	.45**	.03	.06	.05	-.02
10. COMP_PLTFRM	.76	2.99	.65									1	.32**	.17	.22*	-.08	-.29**
11. VARIOUS_CS	.27	2.46	1.03										1	-.09	-.10	.02	.12
12. DESIGN_RDINESS	.54	7.94	2.06											1	.17	.06	-.12
13. Pre-Training Task Score	.82	-	-												1	.39**	-.67**
14. Post-Training Task Score	.82	-	-													1	.42**
15. Design Task Change Score																	1

**p* < .05
***p* < .01

Table 32.

Summary of the Combined Sample: Regression Analysis of Variables of Interest in relation Design Task Performance.[1]

Variable	*B*	*SE B*	β
School	**-2.53**	**.90**	**-.39****
College GPA	.22	.18	.14
Pre-Training Design Task	**.41**	**.09**	**.50****
Prior CS Exp: UNIX Programming	.05	.05	.14
Prior CS Exp: OO Processing	**-.01**	**.03**	**-.09***
Prior CS Exp: Web Designing	.06	.07	.10
Prior CS Exp: Computing Platform	**.13**	**.07**	**.23****
Prior CS Exp: Various CS	.01	.08	.02
Cognitive Ability: Spatial Orientation	-.20	1.34	-.02
Cognitive Ability: Visualization	1.28	.58	.28
Cognitive Ability: Logical Reasoning	.29	.53	.07
Cognitive Ability: Flexibility	-1.83	.65	-.34
Cognitive Ability: Perceptual Style	-.38	.66	-.07
Design Readiness Assessment Scale	-.08	.17	-.05

Notes:
[1] $R^2 = .39$; adjusted $R^2 = .26$; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Research Question Four: Design Readiness Related to OOD Course Performance

The fourth area of inquiry targeted the extent to which the design readiness assessment scale (DRAS) was related to OOD course grade. Linear regression analysis was conducted, showing varying results across the schools and measures of OOD performance (lab grade, project grade, exam grade, and overall course grade).

School A

For School A, it was found that the DRAS was positive and significantly related to project grade ($F(1,74) = 4.43$, $p < .05$, $R^2 = .06$, adjusted $R^2 = .04$) and the overall course grade ($F(1,74) = 6.94$, $p < .01$, $R^2 = .09$, adjusted $R^2 = .07$). The DRAS was not significantly associated with lab grade and exam grade. In summary, the DRAS was able to account for four percent of the variance in project grade and seven percent of the variance in overall grade. A summary of the results is found in Tables 33 – 36.

Table 33.

Summary of the School A: Regression Analysis of Design Readiness Assessment Scale in Relation to Lab Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	1.78	1.23	.17

Notes:
[1] $R^2 = .03$; adjusted $R^2 = .01$; model insignificant (p = .15)
* (p < .05); ** (p < .01)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 34.

Summary of the School A: Regression Analysis of Design Readiness Assessment Scale in Relation to Project Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	**1.83**	**.87**	**.24***

Notes:
[1] R^2 = .06; adjusted R^2 = .04; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 35.

Summary of the School A: Regression Analysis of Design Readiness Assessment Scale in Relation to Exam Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	1.12	.59	.22

Notes:
[1] R^2 = .05; adjusted R^2 = .04; model insignificant ($p = .06$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 36.

Summary of the School A: Regression Analysis of Design Readiness Assessment Scale in Relation to Overall Course Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	**1.40**	**.53**	**.29***

Notes:
[1] $R^2 = .09$; adjusted $R^2 = .07$; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

School B

For School B, The DRAS was not significantly associated with any measure of course performance—lab grade, project grade, exam grade, or overall course grade. Tables 37 – 40 show the results of these analyses.

Table 37.

Summary of the School B: Regression Analysis of Design Readiness Assessment Scale in Relation to Lab Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	1.23	.70	.19

Notes:
[1] $R^2 = .04$; adjusted $R^2 = .02$; model insignificant ($p = .09$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 38.

Summary of the School B: Regression Analysis of Design Readiness Assessment Scale in Relation to Project Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	-.12	.55	-.03

Notes:
[1] R^2 = .00; adjusted R^2 = -.01; model insignificant (p = .82)
* (p < .05); ** (p < .01)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 39.

Summary of the School B: Regression Analysis of Design Readiness Assessment Scale in Relation to Exam Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	.64	.45	.15

Notes:
[1] R^2 = .02; adjusted R^2 = .01; model insignificant (p = .16)
* (p < .05); ** (p < .01)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 40.

Summary of the School B: Regression Analysis of Design Readiness Assessment Scale in Relation to Overall Course Grade.[1]

Variable	*B*	*SE B*	β
Design Readiness Assessment Scale	.42	.41	.11

Notes:
[1] R^2 = .01; adjusted R^2 = .00; model insignificant (p = .31)
* (p < .05); ** (p < .01)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Combined Sample

For the combined sample, the variable *school* was introduced into the regression model as a covariant. The DRAS was positively associated with three measures of course performance: lab grade ($F(2,158) = 4.49$, $p < .01$, $R^2 = .05$, adjusted $R^2 = .04$) , exam grade ($F(2,158) = 20.04$, $p < .05$, $R^2 = .20$, adjusted $R^2 = .19$) , and the overall course grade ($F(2,158) = 3.82$, $p < .05$, $R^2 = .05$, adjusted $R^2 = .04$). It was found that the DRAS was not significantly related to project grade; however school showed a positive and significant relationship to project grade. School was also shown to be positive and significantly related to exam grade. In summary, the results show that the DRAS was able to account for four percent of the variance in lab and overall grade and nineteen percent of the variance in exam grade. A summary of the results is found in Tables 41 – 44.

Table 41.

Summary of the Combined Sample: Regression Analysis of Design Readiness Assessment Scale in Relation to Lab Grade.[1]

Variable	*B*	*SE B*	β
School	2.98	2.82	.09
Design Readiness Assessment Scale	**1.49**	**.68**	**.18****

Notes:
[1] $R^2 = .05$; adjusted $R^2 = .04$; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 42.

Summary of the Combined Sample: Regression Analysis of Design Readiness Assessment Scale in Relation to Project Grade.[1]

Variable	*B*	*SE B*	β
School	**5.13**	**2.08**	**.20***
Design Readiness Assessment Scale	.80	.51	.13

Notes:
[1] $R^2 = .08$; adjusted $R^2 = .07$; model significant ($p < .01$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Table 43.

Summary of the Combined Sample: Regression Analysis of Design Readiness Assessment Scale in Relation to Exam Grade.[1]

Variable	*B*	*SE B*	β
School	**6.73**	**1.51**	**.35****
Design Readiness Assessment Scale	**.89**	**.37**	**.18***

Notes:
[1] $R^2 = .20$; adjusted $R^2 = .19$; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)

Table 44.

Summary of the Combined Sample: Regression Analysis of Design Readiness Assessment Scale in Relation to Overall Course Grade.[1]

Variable	*B*	*SE B*	β
School	-.42	1.36	-.03
Design Readiness Assessment Scale	**.88**	**.33**	**.22****

Notes:
[1] $R^2 = .05$; adjusted $R^2 = .04$; model significant ($p < .05$)
* ($p < .05$); ** ($p < .01$)
(*B*): the unstandardized beta coefficient
(*SE B*): unstandardized coefficient standard error
(β): standardized beta coefficient

Research Question Five: Design Readiness Related to Design Task Performance

The final area of inquiry targeted the extent to which design readiness was related to performance on the post-training design task. Linear regression analysis was conducted using the

DRAS and post-training design task score. The pre-training design task was entered into the model as a covariant. It was found that the DRAS was not significantly associated with post-training design task performance for School A, School B, or the Combined Sample.

Path Analysis

The final section of this chapter is aimed at characterizing the network of relationships between variables of interest and the dependent variables of course outcomes (lab grade, project grade, exam grade, and overall grade) and design task performance. The linear regressions summarized in previous sections were used as the foundation for a series of path analyses that explored the relation of exogenous variables such as college grade point average, course grade and cognitive abilities, the intermediate variable the design readiness, and outcome (or endogenous) variables such as course grade and design task performance.

The use of structural equation modeling was considered as a more precise analysis technique for this study. However, due to the number of instruments and the final number of participants the associated statistical tests suffered from a lack of statistical power. Furthermore because the research is an exploratory study and that has been guided by general research questions rather than specific hypotheses, path analysis is an appropriate technique for constructing and presenting models that are not intended to show causal relationships between variables. Instead the links discovered in the analysis simply document the pattern of relationships or associations among variables as determined by overlapping regression analysis. Path coefficients can be estimated by ordinary least squares regression (Billings & Wroten, 1978, Pedhazur 1982). This procedure is advantageous because the parameters are estimated for each equation separately (Dillon & Goldstein, 1984). In order to perform path analyses, the following four statistical assumptions must be met:

1. The relationships of interest should be formulated as a set of linear, additive and causal relationships among the variables (Pedhazur, 1982).
2. Linear regression (Tait & Vessey, 1989) should show no evidence of reciprocal causation among the variables. The present data shows evidence of multicollinearity among the variables and may violate this assumption. The correlation tables revealed multicollinearity among all the cognitive measures, as well as the prior computer science experience scales.
3. The residuals of the endogenous variables should not correlate significantly with any of the preceding variables in the model (Pedhazur, 1982).
4. Variables should be measured on an interval scale. Although one variable for this study was not measured on an interval scale (*school*), this violation should not negatively impact the outcome. Indeed, some experts have maintained that the use of dummy coding may be considered to be a sufficient remedy in such a case (Billings & Wroten, 1978).

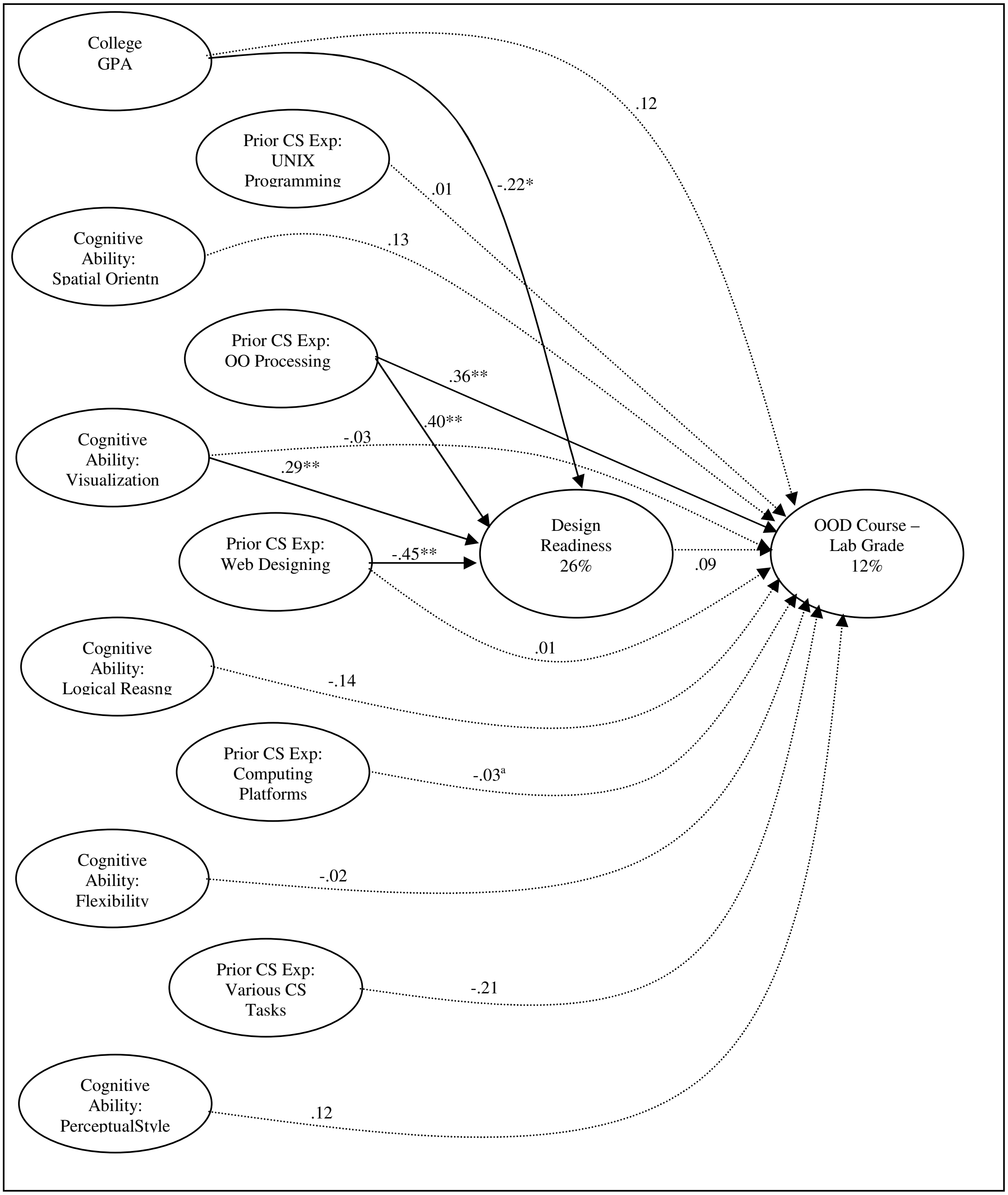
Typically, path analysis uses the terminology *direct/indirect effect* to denote relationships between variables. However, due to the exploratory nature of this study, the relationships are instead explained in terms of *direct/indirect relationships*. Note that as with the regression analyses in general, the path models were constructed separately for School A, School B and the combined sample.

Course Grade – School A

The relationship between the exogenous variables and the endogenous variables (lab grades, project grades, exam grade, and overall grade) was represented with standardized beta coefficients on the significant links.

Lab Grade

An inspection of Figure 4 reveals only one variable with a relationship to the lab grades for School A. The prior computer science experience object-oriented processing ($\beta = .36, p < .01$) revealed a significant direct relationship with the lab grade. Although the prior computer science experiences object-oriented processing ($\beta = .40, p < .01$) and web designing ($\beta = -.45, p < .01$), the cognitive ability visualization ($\beta = .29, p < .01$), and college grade point average ($\beta = -.22, p < .01$) showed a significant direct relationship to the design readiness assessment scale (DRAS), the DRAS was not significantly related to lab grade. Therefore, there were no indirect relationships to lab grade. The model accounted for twelve percent of the variance in lab grade.



* (p < .05); ** (p < .01); [a](p < .07)

Figure 4.

Path Analysis Model for SCHOOL A – Lab Grade.

Project Grade

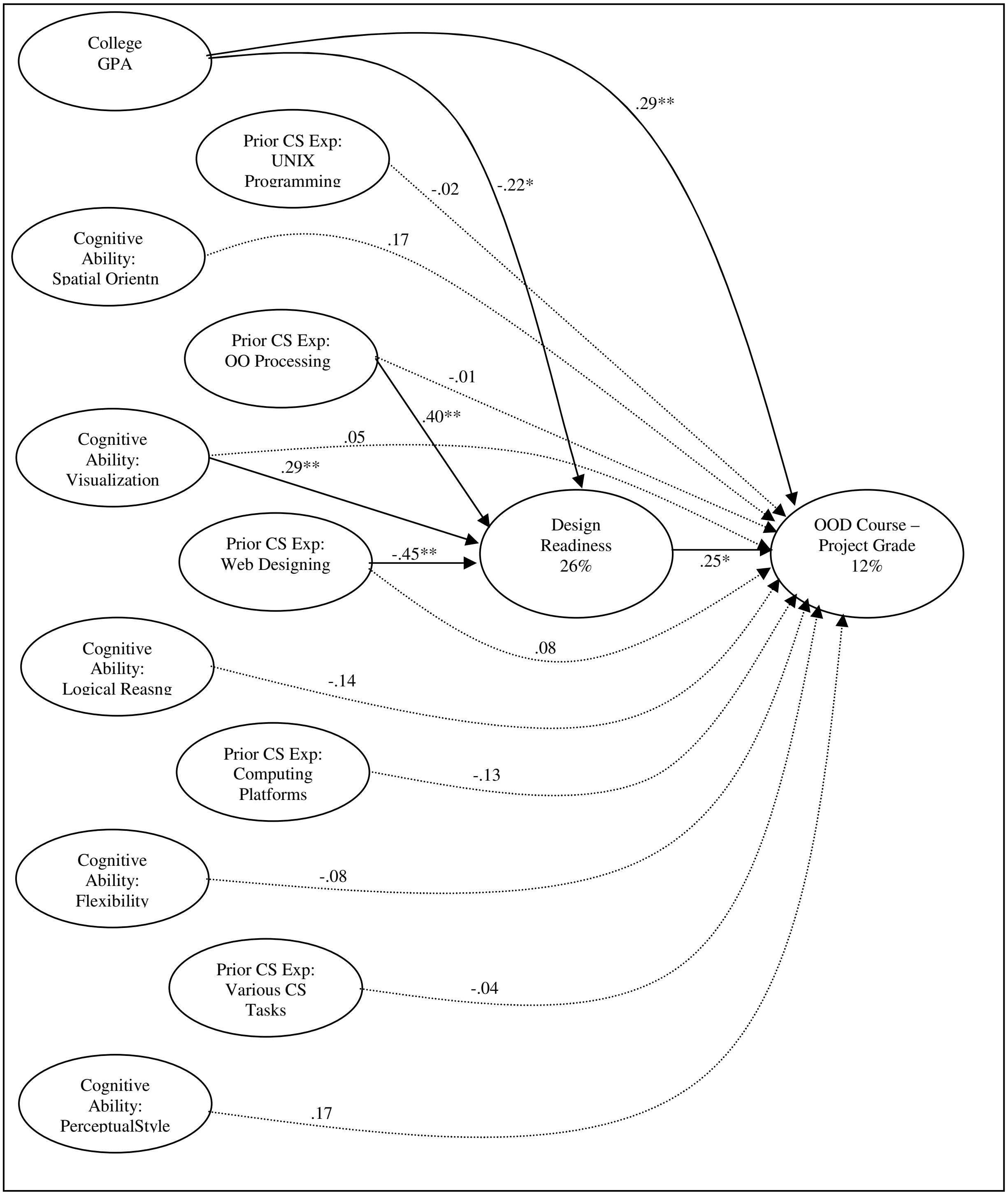
An inspection of Figure 5 reveals two variables with significant relationships to project grades for School A. The design readiness assessment scale ($\beta = .25$, $p < .01$) and college grade point average ($\beta = .29$, $p < .05$) showed a significant direct relationship with project grade. The prior computer science experiences object-oriented processing ($\beta = .40$, $p < .01$) and web designing ($\beta = -.45$, $p < .01$), the cognitive ability visualization ($\beta = .29$, $p < .01$), and college grade point average ($\beta = -.22$, $p < .01$) show significant indirect relationships with project grades through the design readiness assessment scale. The model explained twelve percent of the variation in OOD course grade.

Exam Grade

An inspection of Figure 6 reveals two variables with relationships to exam grade for School A. The cognitive ability-visualization ($\beta = .40$, $p < .01$) and college grade point average ($\beta = .32$, $p < .01$) showed a significant direct relationships with the exam grade. Because the design readiness assessment scale was not significantly related to exam grade, there were no significant indirect relationships through design readiness. The model explained twenty-eight percent of the variation in OOD course grade.

Overall Course Grade

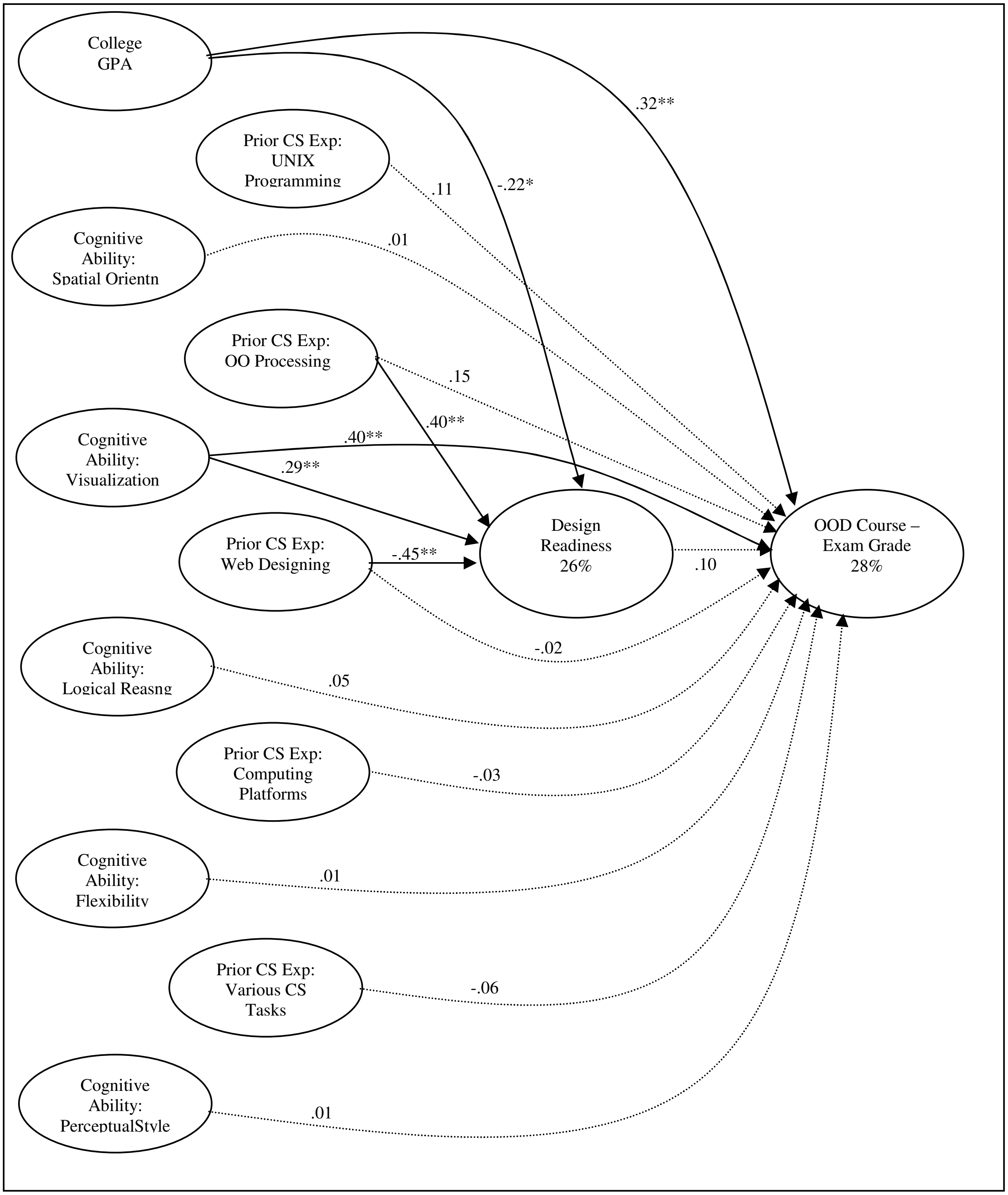
Inspection of Figure 7 reveals three variables with relationships to overall course grade for School A. The design readiness assessment scale ($\beta = .22$, $p < .05$), college grade point average ($\beta = .29$, $p < .05$), and visualization ($\beta = .26$, $p < .01$) showed significant direct relationships with the overall course grade. The prior computer science experiences object-oriented processing ($\beta = .40$, $p < .01$) and web designing ($\beta = -.45$, $p < .01$), the cognitive ability visualization ($\beta = .29$, $p < .01$), and college grade point average ($\beta = -.22$, $p < .01$) show significant indirect relationships through the mediating variable design readiness. The model explained thirty percent of the variation in OOD overall course grade.



* (p < .05); ** (p < .01)

Figure 5.

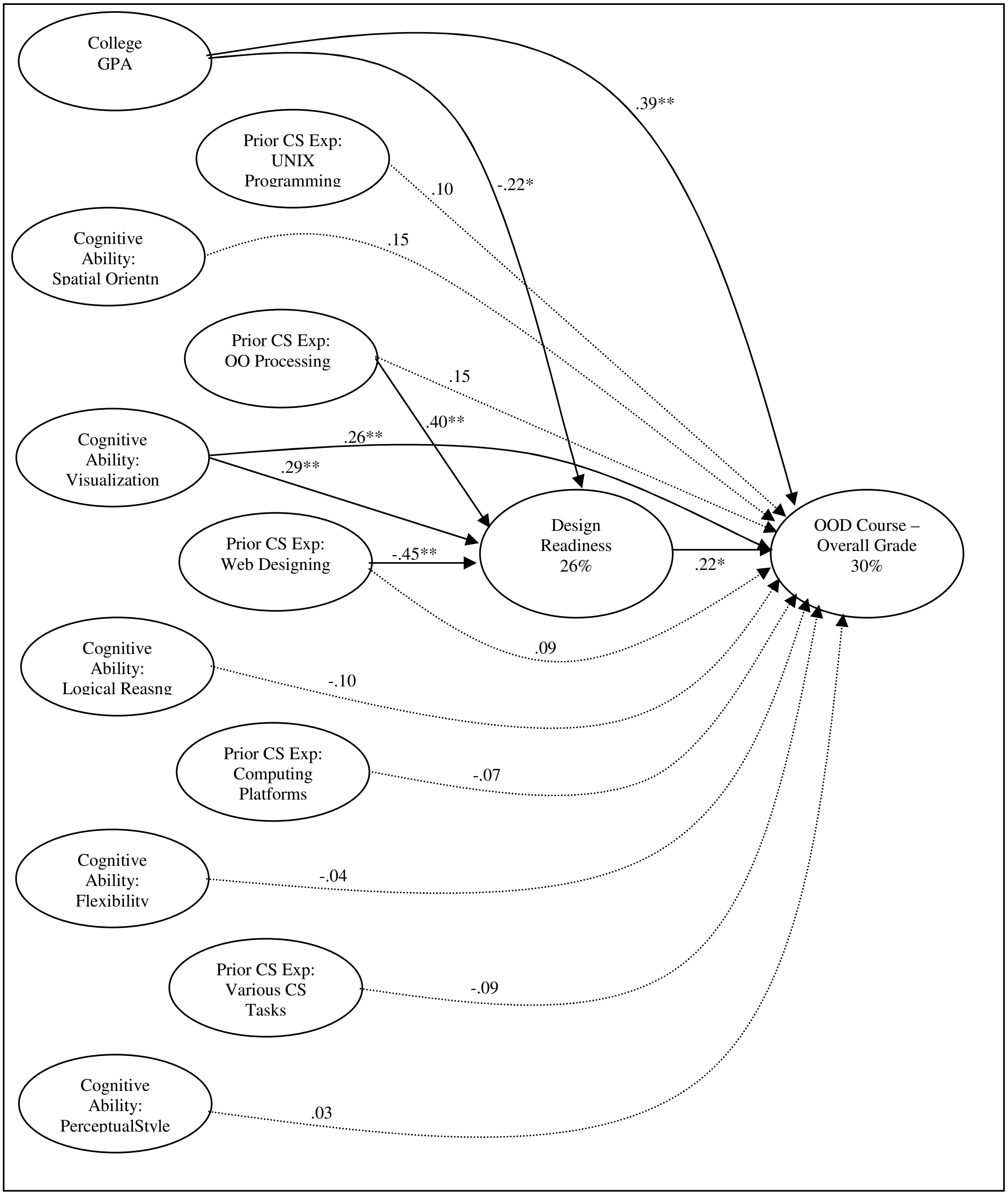
Path Analysis Model for SCHOOL A –Project Grade.



* (p < .05); ** (p < .01)

Figure 6.

Path Analysis Model for SCHOOL A –Exam Grade.



* (p < .05); ** (p < .01)

Figure 7.

Path Analysis Model for SCHOOL A –Overall Grade.

Course Grade – School B

The relationship between the exogenous variables and the endogenous variables (lab grades, project grades, exam grade, and overall grade) were represented with standardized beta coefficients on the significant links.

Lab Grade

As shown in Figure 8, the relationships between exogenous and endogenous variables did not lead to a significant path model for School B with regard to lab grade.

Project Grade

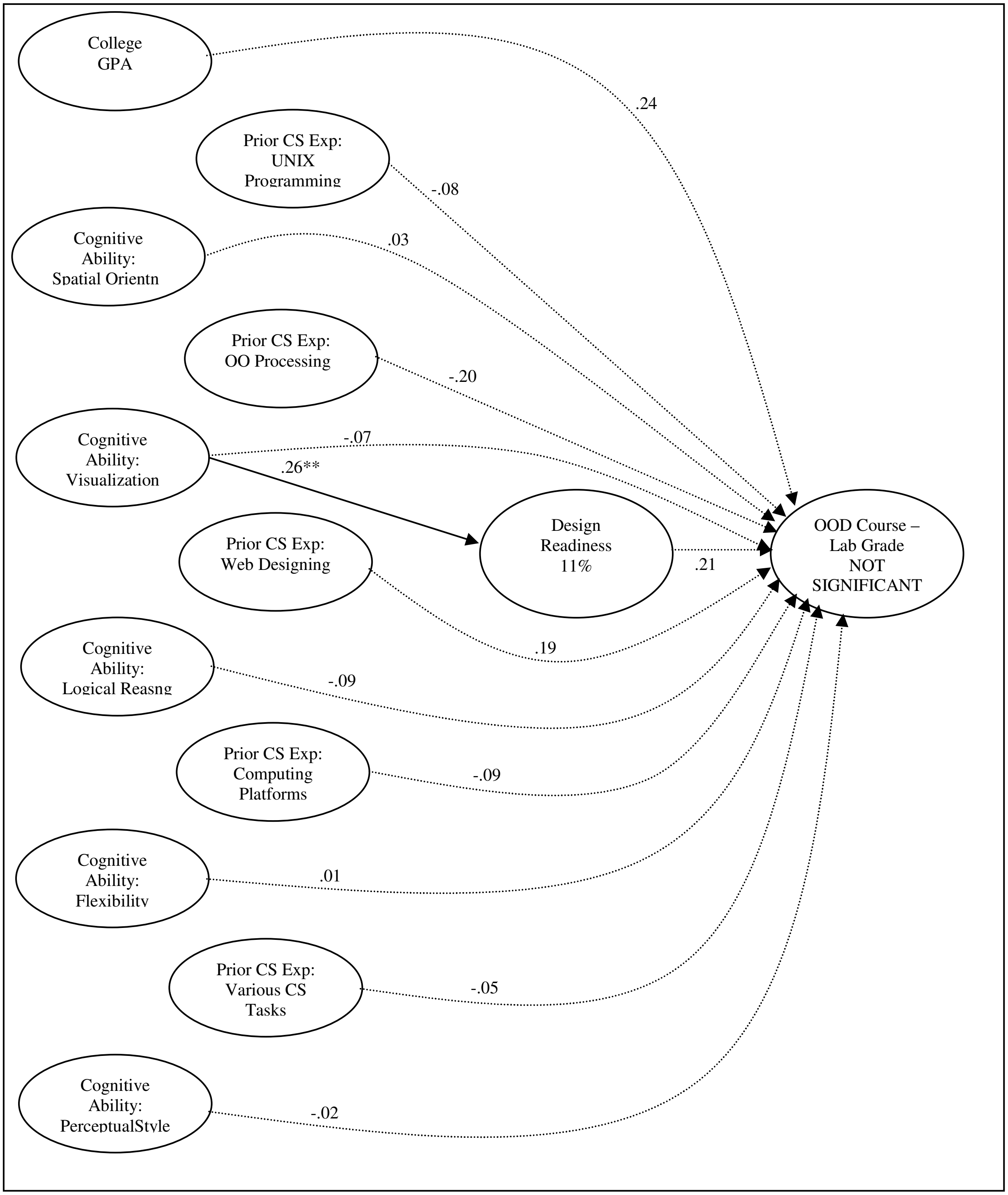
An inspection of Figure 9 reveals two variables with relationships to project grade for School B. The college grade point average ($\beta = .41, p < .01$) and the cognitive ability measure of logical reasoning ($\beta = -.27, p < .01$) showed significant direct relationships with the project grade, but there were no direct or indirect relations through the variable of design readiness. The model explained twenty-one percent of the variation in project grades for this school.

Exam Grade

An inspection of Figure 10 reveals only one variable with a relationship to exam grade for School B. College grade point average ($\beta = .42, p < .01$) showed a significant direct relationship with the exam grade, but there were no direct or indirect relations through the variable of design readiness. The model explained seventeen percent of the variation in OOD exam grades.

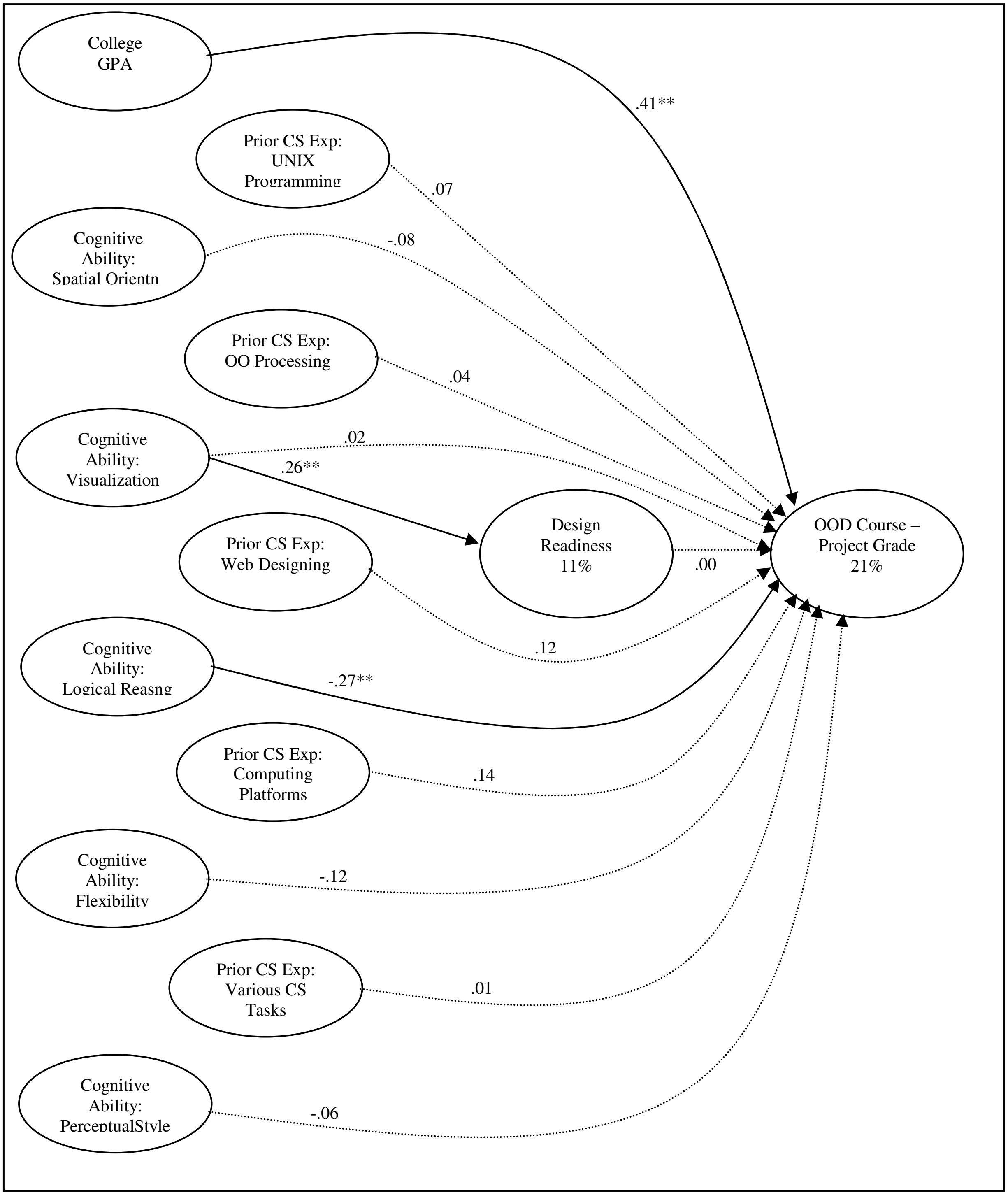
Overall Grade

An inspection of Figure 11 reveals only one variable with a relationship to overall course grade for School B. College grade point average ($\beta = .49, p < .05$) showed a significant direct relationship with the overall course grade, but there were no direct or indirect relations through the variable of design readiness. The model explained twenty-three percent of the variation in OOD overall course grade.



* (p < .05); ** (p < .01)
Figure 8.

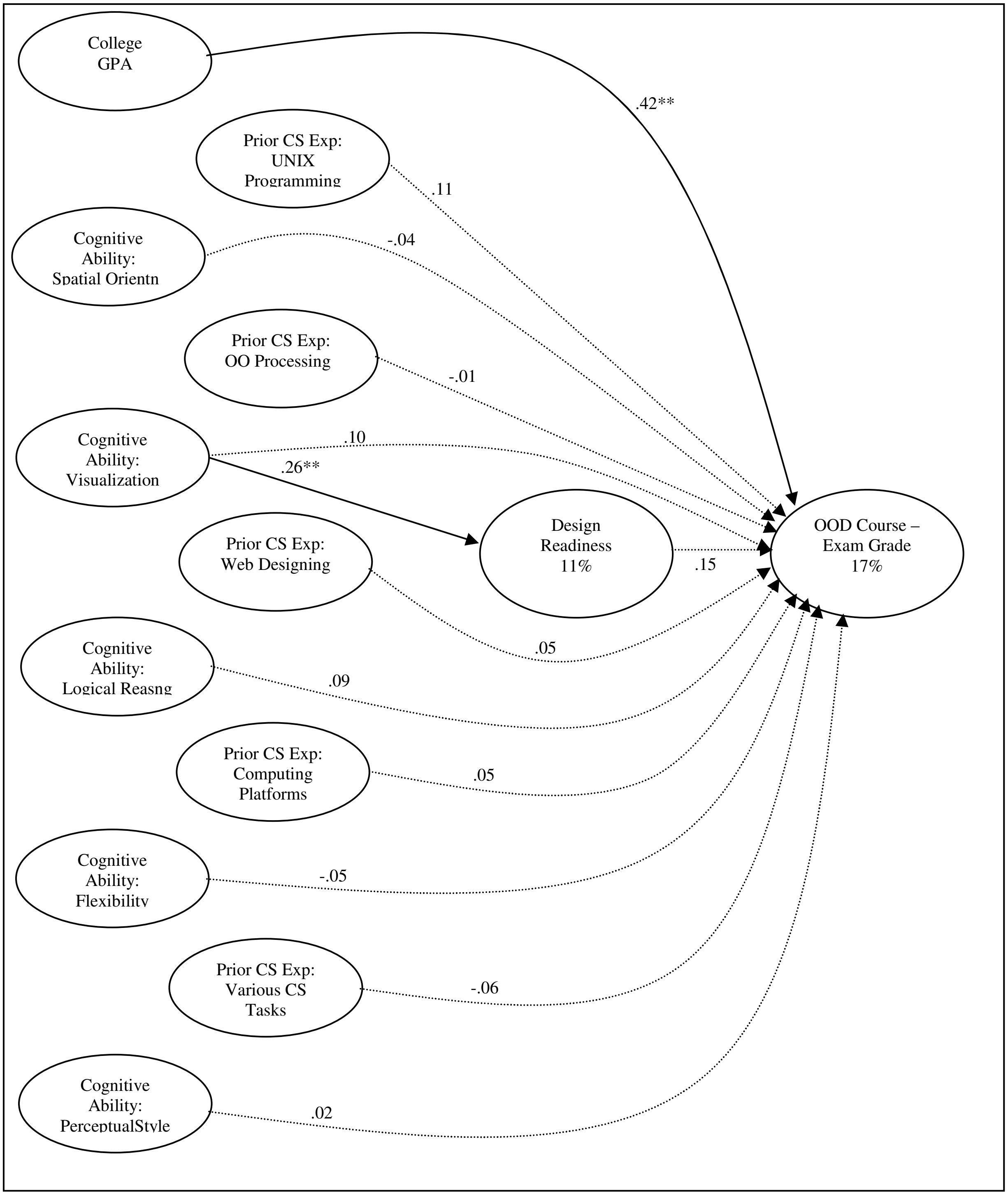
Path Analysis Model for SCHOOL B – Lab Grade.



* (p < .05); ** (p < .01)

Figure 9.

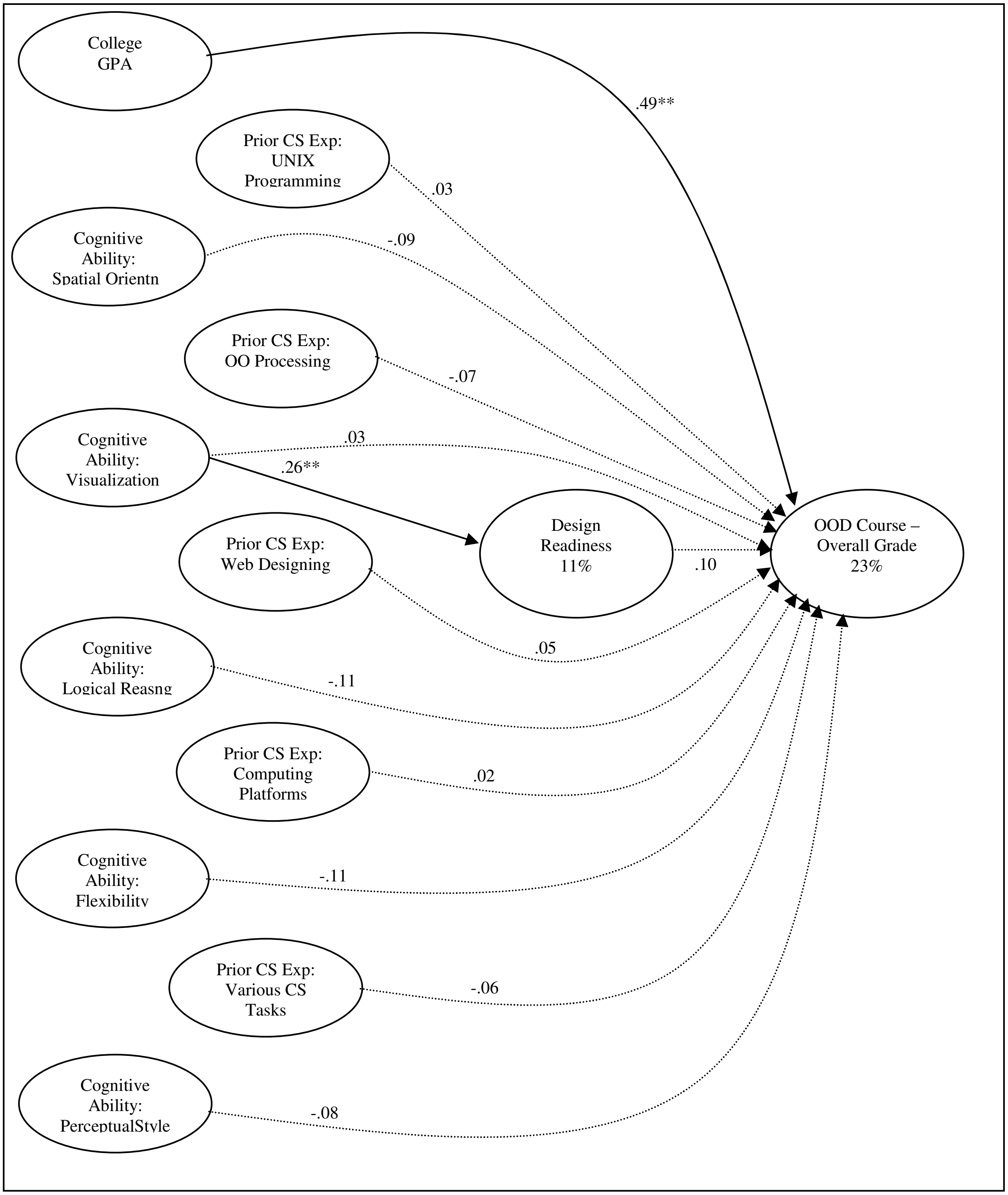
Path Analysis Model for SCHOOL B – Project Grade.



* (p < .05); ** (p < .01)

Figure 10.

Path Analysis Model for SCHOOL B – Exam Grade.



* (p < .05); ** (p < .01)

Figure 11.

Path Analysis Model for SCHOOL B – Overall Grade.

Course Grade – Combined Sample

The relationship between the exogenous variables and the endogenous variables (lab grades, project grades, exam grade, and overall grade) were represented with standardized beta coefficients on the significant links.

Lab Grade

An inspection of Figure 12 reveals four variables with relationships to the combined sample – lab grade. The college grade point average ($\beta = .24$, $p < .01$), the prior computer science experience measure – web designing ($\beta = .20$, $p < .01$), and the prior computer science experience measure – various CS ($\beta = -.18$, $p < .05$), and the design readiness assessment scale ($\beta = .20$, $p < .01$) showed positive and significant direct relationships with the lab grade. School and the cognitive ability measure of visualization were also seen to have a positive and significant indirect relationship with lab grade through design readiness. Recall that design readiness was a significant predictor of course outcomes only for School A. The model explained fourteen percent of the variation in the combined sample lab grade.

Project Grade

An inspection of Figure 13 reveals two variables with relationships to project grade for School B. College grade point average ($\beta = .34$, $p < .01$) and school ($\beta = .21$, $p < .01$) showed a significant direct relationship with the project grade, but there were no direct or indirect relations through the variable of design readiness. The model explained fifteen percent of the variation in the combined sample of project grades.

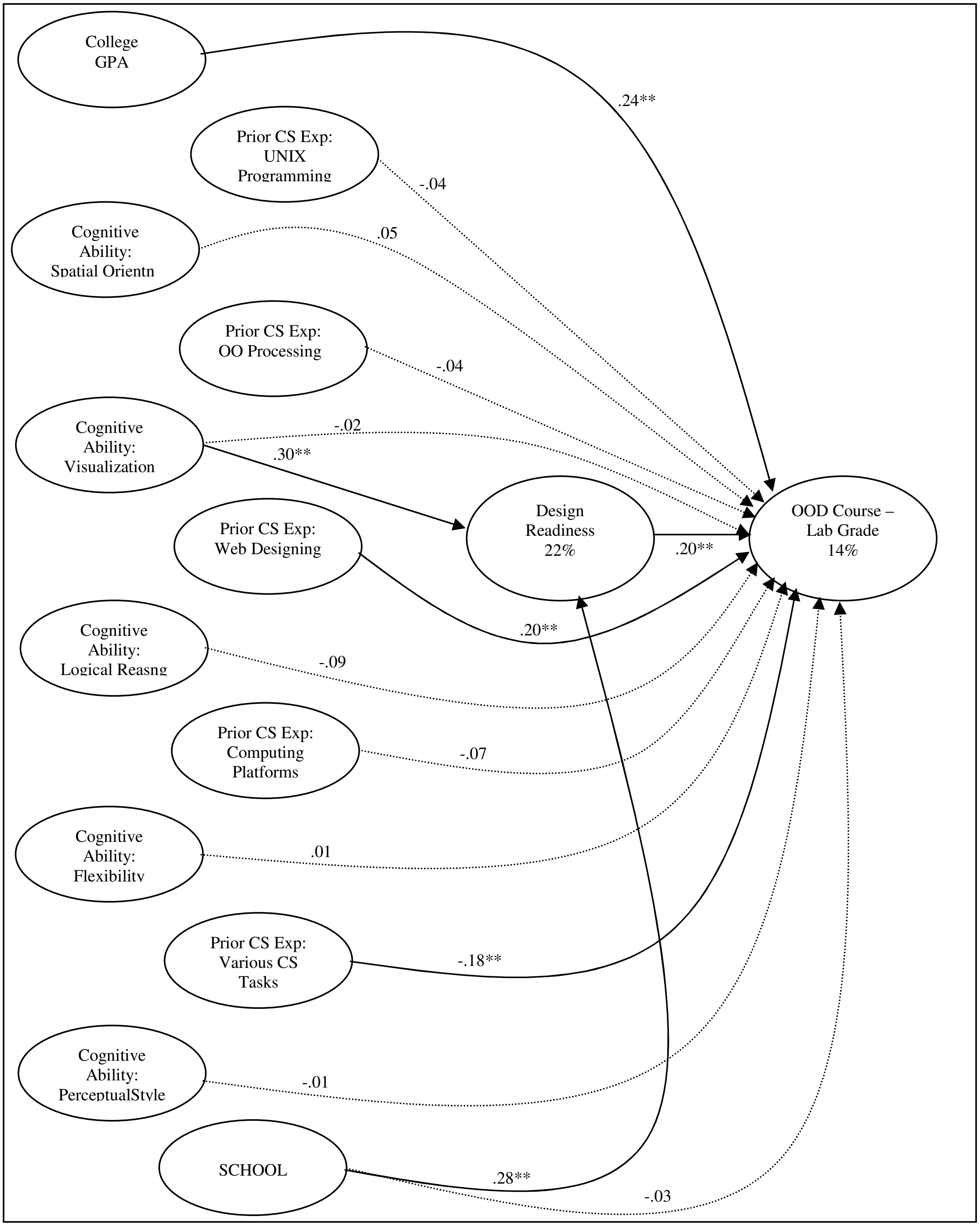
Exam Grade

Inspection of Figure 14 reveals three variables with relationships to exam grade for School B. The cognitive ability - visualization ($\beta = .28$, $p < .01$), college grade point average ($\beta =$

.34, $p < .01$), and school ($\beta = .27$, $p < .01$) showed a significant direct relationship with the exam grade, but there were no direct or indirect relations through the variable of design readiness. The model explained thirty-five percent of variation in the combined sample exam grade.

Overall Grade

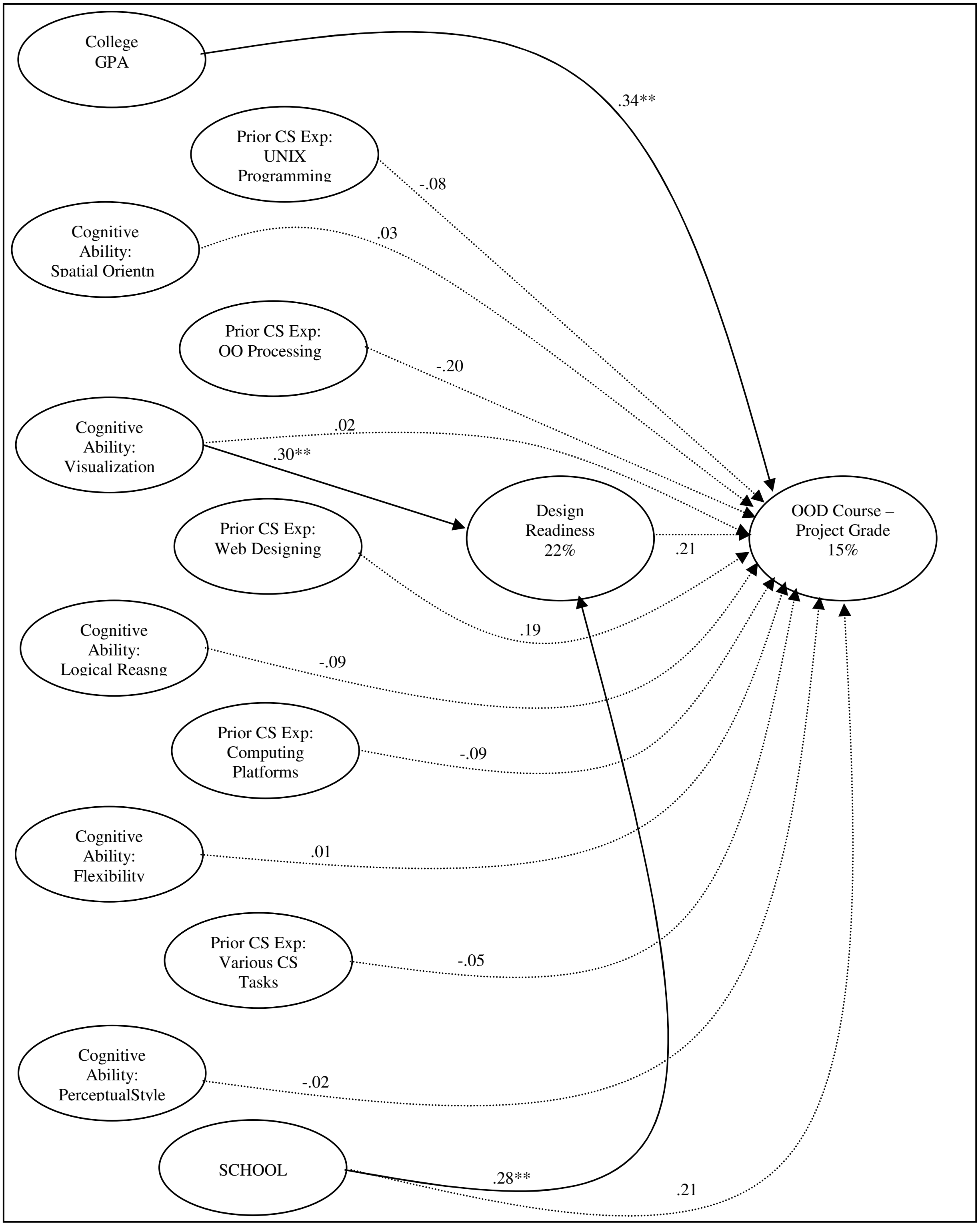
An inspection of Figure 15 reveals two variables with a direct relationship to the final course grade for the combined sample. College grade point average ($\beta = .45$, $p < .01$) and design readiness ($\beta = .19$, $p < .01$) had significant and positive relations with overall grade in the combined sample of students. Furthermore, the cognitive ability visualization ($\beta = .50$, $p < .01$), and the school variable ($\beta = .28$, $p < .01$) were shown to have significant and positive indirect relationships with final grade, through the mediating variable of design readiness. These findings are quite promising for the future of the DRAS. The researcher is still mindful of that Kehoe (2000) stated that important decisions concerning individual students should not be based on a single test score when the reliability rating was less than .80. Despite the fact of the low reliability, the DRAS was shown to be significantly related to overall course performance. A more reliable version may serve as an appropriate tool to identify students for proper course placement.



* (p < .05); ** (p < .01)

Figure 12.

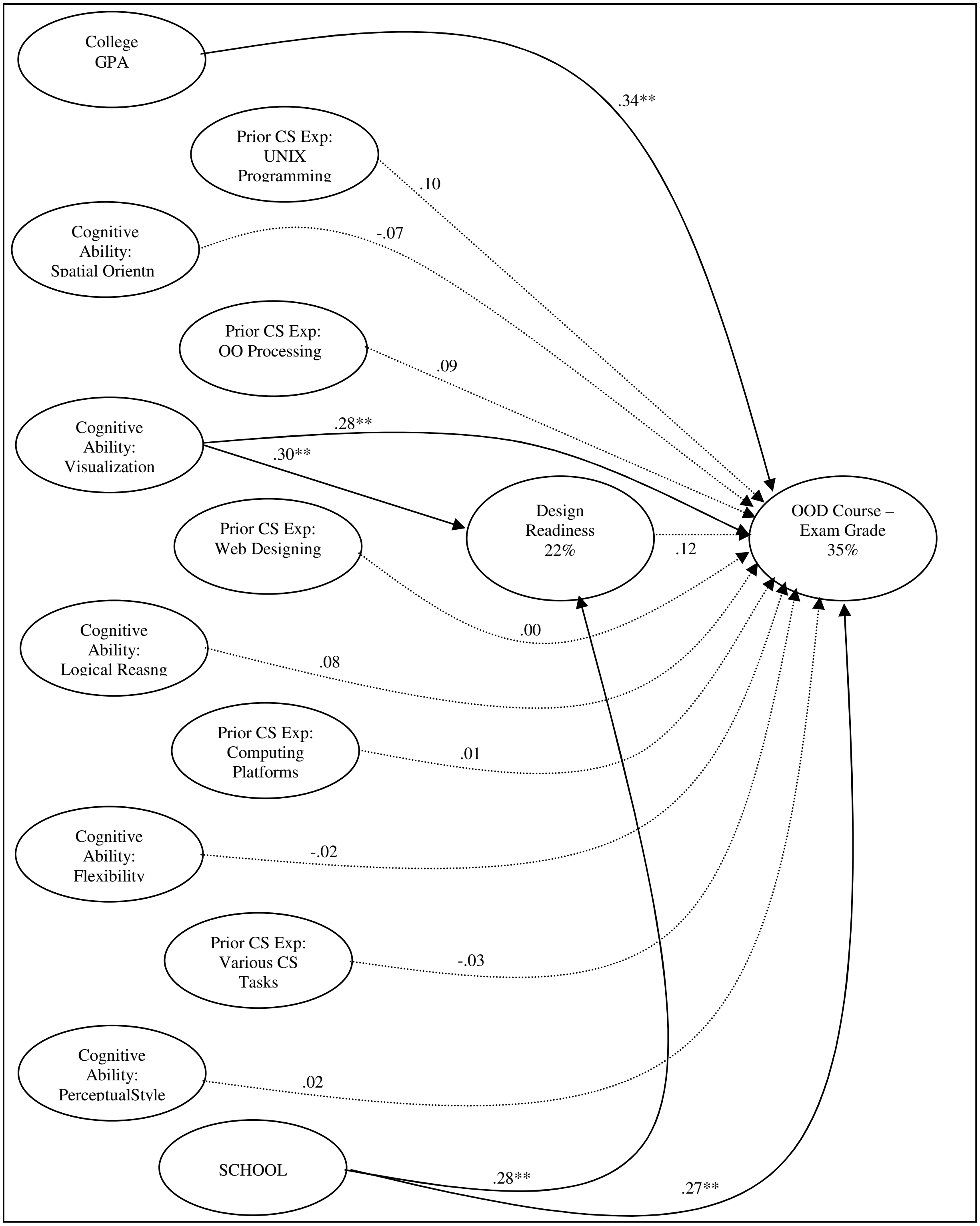
Path Analysis Model for COMBINED SAMPLE– Lab Grade.



* (p < .05); ** (p < .01)

Figure 13.

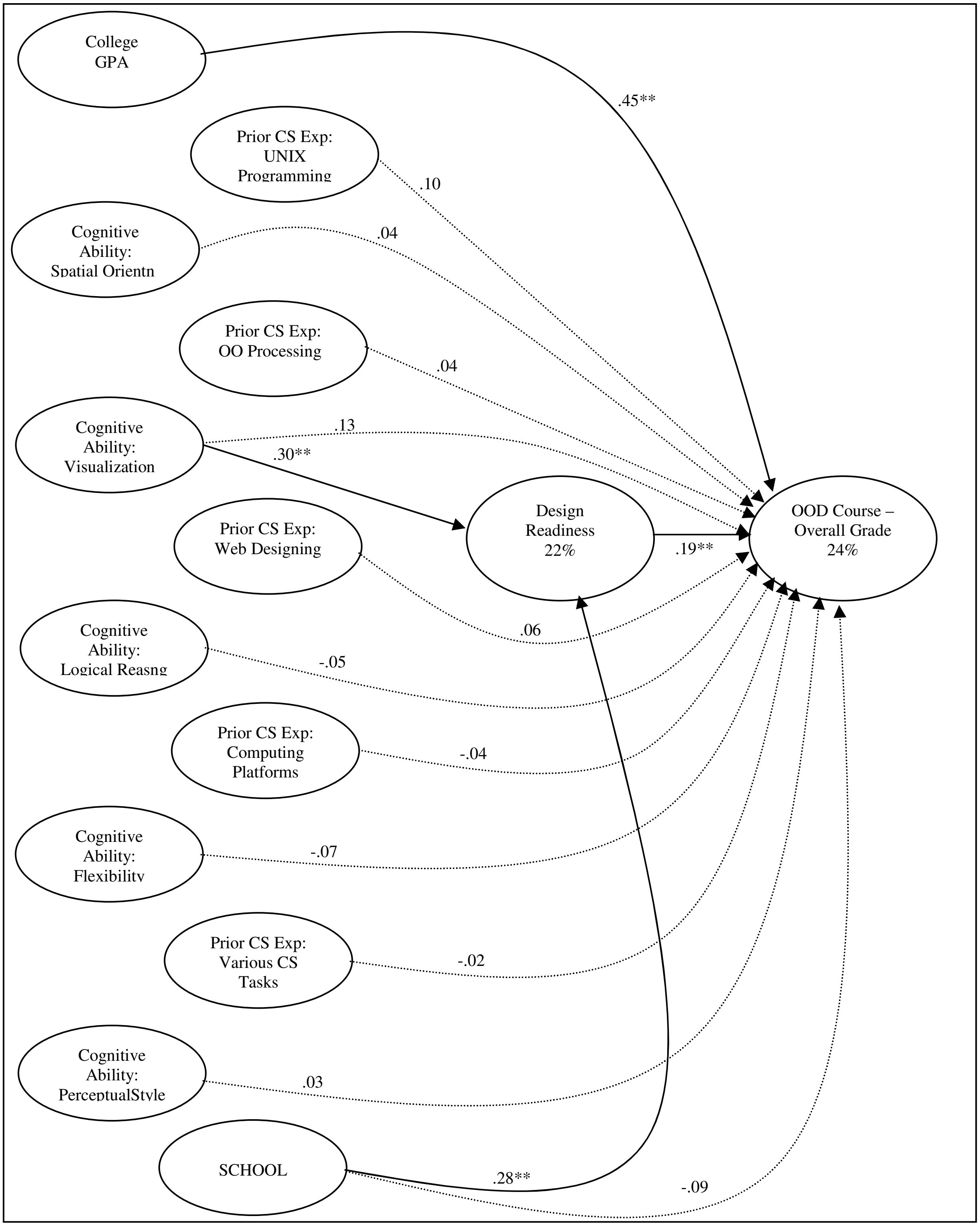
Path Analysis Model for COMBINED SAMPLE – Project Grade.



* (p < .05); ** (p < .01)

Figure 14.

Path Analysis Model for COMBINED SAMPLE – Exam Grade.



* (p < .05); ** (p < .01)

Figure 15.

Path Analysis Model for COMBINED SAMPLE – Overall Grade.

Design Task – School A

The relationships between the various exogenous variables and the endogenous variable of post-training design task were represented with standardized beta coefficients on the significant links. Figure 16 shows a summary of the relationships explored in this path analysis.

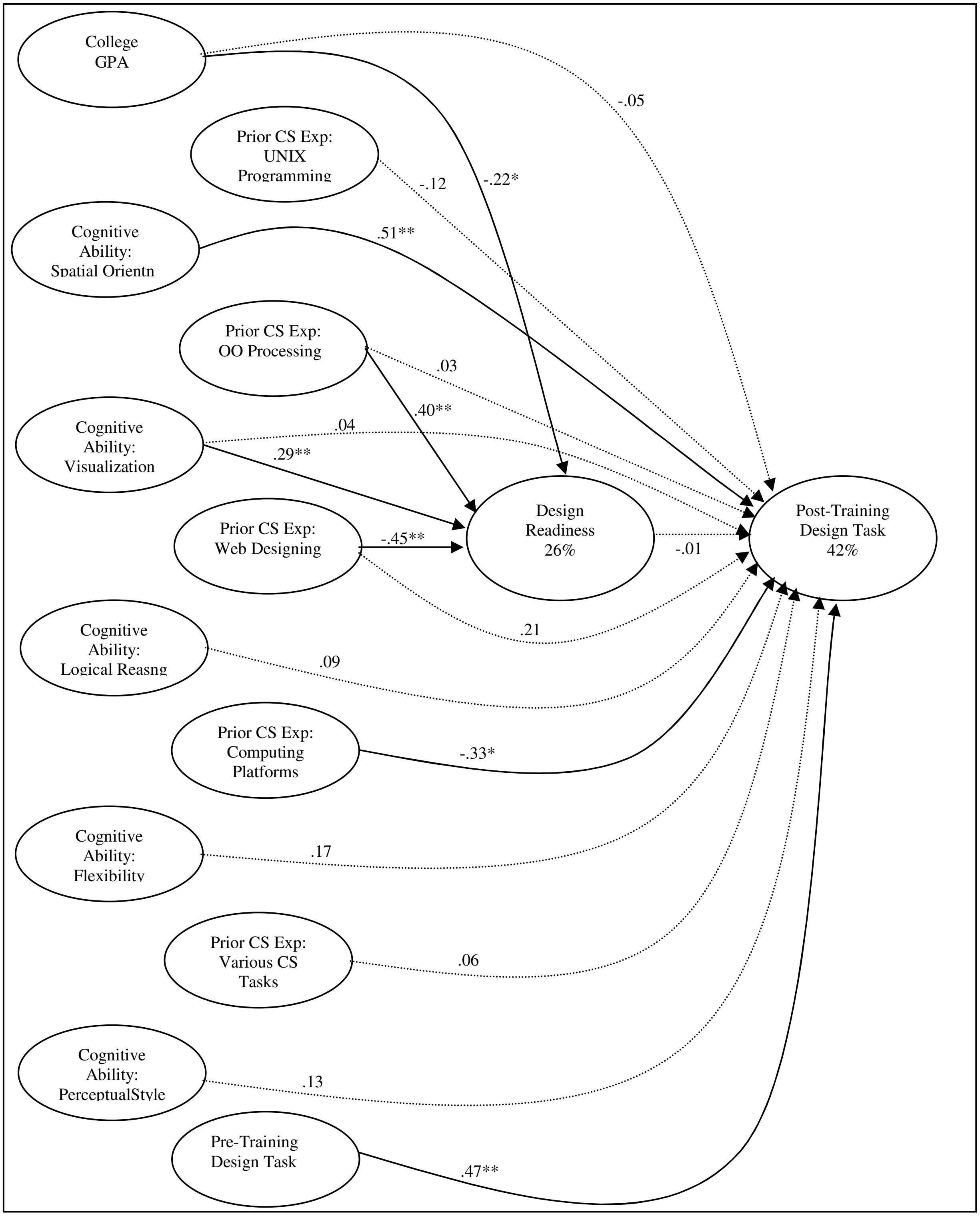
When examining the relationships between various measures of individual differences and the post-training design task score, for School A, it was found that two individual difference measures and the pre-training design task showed significant direct relationships. The cognitive ability spatial orientation ($\beta = .51$, $p < .01$) and pre-training design task score ($\beta = .47$ $p < .01$) were positive and significantly related to post-training design task score; the prior computer science experience computing platforms ($\beta = -.33$, $p < .05$) was negative and significantly related to post-training design task score. There were no significant indirect relationships. It was also noted that the relation of logical reasoning and post-training design task approached significance ($\beta = .10$, $p < .07$). The model explained forty-two percent of the variance in post-training design task score.

Design Task – School B

The standardized path coefficients for School B are shown in Figure 17. The relationship between the exogenous variables and the endogenous variable — post-training design task score — are represented with standardized beta coefficients on the significant links. Inspection of Figure 16 revealed only one individual difference variable with a relationship to post-training design task. Along with pre-training design task, object-oriented processing ($\beta = .33$, $p < .01$), showed a positive and significant direct relationship with the post-training design task score. The model explained twenty-three percent of the variance in post-training design task score.

Design Task – Combined Sample

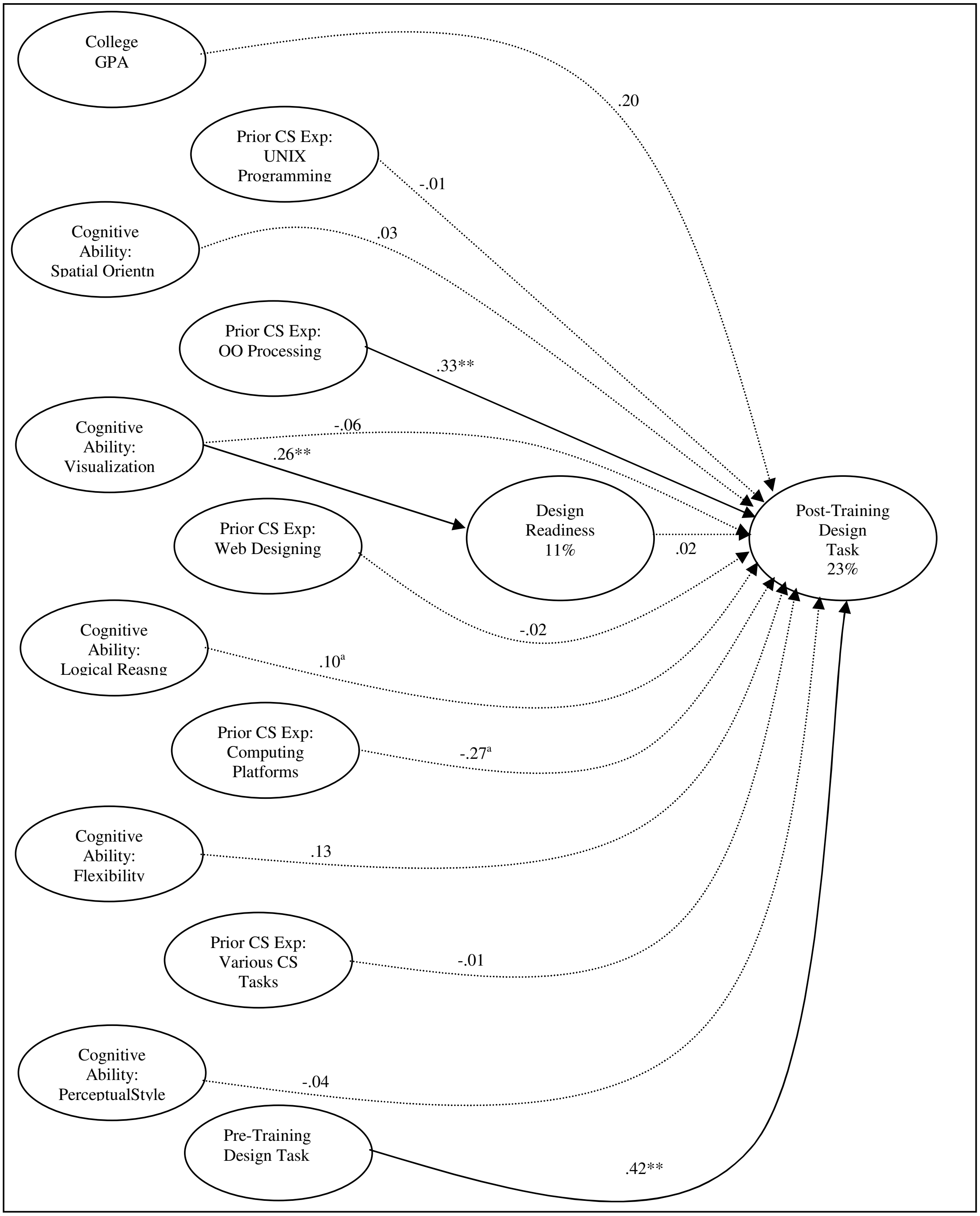
The standardized path coefficients for the combined sample are shown in Figure 18. Inspection of Figure 18 showed that pre-training design task score was the only variable with a positive significant relationship with the post-training design task. There were no significant negative relationships found. It was found that the relation of college grade point average and post-training design task approached significance ($\beta = .18$, $p < .08$). The model explained fourteen percent of the variance in the post-training design task score.



* (p < .05); ** (p < .01)

Figure 16.

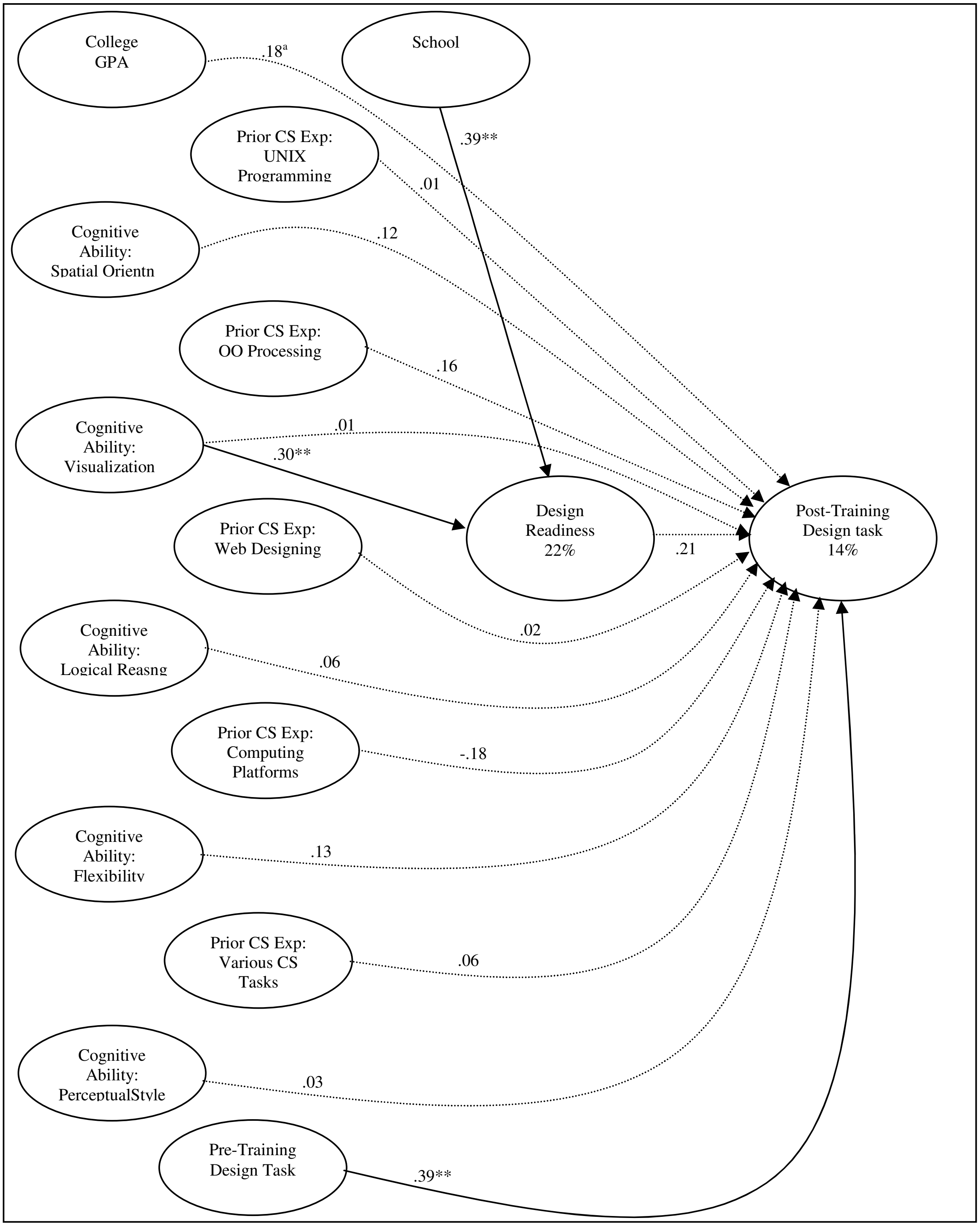
Path Analysis Model for DESIGN TASK – SCHOOL A.



* (p < .05); ** (p < .01); [a] (p < .07)

Figure 17.

Path Analysis Model for DESIGN TASK – SCHOOL B.



* (p < .05); ** (p < .01); [a] (p <.08)

Figure 18.

Path Analysis Model for DESIGN TASK – COMBINED.

A summary of the significant results of this research is presented according to the research questions in the following section.

Summary of Research Questions

Table 45 summarizes the results of the analyses conducted to investigate each research question. As shown in this table, the outcomes vary schools and combined sample. Significant direct (D) and indirect (I) relationships are shown for each measure of performance – course and design task; as well as, whether the relationship was positive (+) or negative (-).

Table 45.

Summary of Research Questions

	Research Questions	Significant Results		
		SCHOOL A	SCHOOL B	Combined Sample
1.	What student characteristics are related to design readiness?	• Visualization (+) • OO Processing (+) • Web Designing (-) • College GPA (-)	• Visualization (+)	• School (+) • Visualization (+)
2.	What student characteristics are related to performance in an OOD course?	• Lab Grade ○ OO Processing (+)(D) • Project Grade ○ Design Readiness (+)(D) ○ College GPA (+)(D) ○ Visualization (+)(I) ○ OO Processing (+)(I) ○ Web Designing (-)(I) ○ College GPA (-)(I) • Exam Grade ○ Visualization (+)(D) ○ College GPA (+)(D) • Overall Grade ○ Visualization (+)(D) ○ College GPA (+)(D) ○ Design Readiness (+)(D) ○ Visualization (+)(I) ○ OO Processing (+)(I) ○ Web Designing (-)(I) ○ College GPA (-)(I)	• Lab Grade ○ None • Project Grade ○ Logical Reasoning (-)(D) ○ College GPA (+)(D) • Exam Grade ○ College GPA (+)(D) • Overall Grade ○ College GPA (+)(D)	• Lab Grade ○ College GPA (+)(D) ○ Web Designing (+)(D) ○ Design Readiness (+)(D) ○ Various CS (-)(D) ○ School (+) (I) ○ Visualization (+)(I) • Project Grade ○ School (+)(D) ○ College GPA (+)(D) • Exam Grade ○ School (+)(D) ○ College GPA (+)(D) ○ Visualization (+)(D) • Overall Grade ○ Design Readiness (+)(D) ○ College GPA (+)(D) ○ School (+) (I) ○ Visualization (+)(I)
3.	What student characteristics are related to performance on a design task?	• Pre-training design task (+)(D) • Spatial Orientation (+)(D) • Computing Platforms (+)(D)	• Pre-training design task (+)(D) • OO Processing (+)(D)	• Pre-training design task (+)(D)
4.	Is design readiness related to a student's performance in an OOD course?	Yes	No	Yes
5.	Is design readiness related to a student's performance on a design task?	No	No	No

(+): Positive standardized beta coefficient
(-): Negative standardized beta coefficient
(D): direct relationship
(I): Indirect relationship

Research Question One: Student Characteristics Related to Design Readiness

It was found that the cognitive ability visualization was positively associated with the design readiness assessment scale for both schools, as well as the combined sample. For School A, the prior computer science experience measure of OO processing was positively related to the design readiness assessment scale. Further, the prior computer science experience measure of web designing and the college grade point average were negatively related to the design readiness assessment scale. Visualization was the only individual related to the design readiness assessment scale for School B. When examining the Combined Sample, that along with visualization, School was also positively related to the design readiness assessment scale.

Research Question Two: Student Characteristics Related to OOD Course Performance

The results varied across school and OOD course performance measure (lab grade, project grade, exam grade, and overall grade).

For School A, lab grades were found to have a direct positive relationship to OO processing. Further, project grades had direct positive relationships with design readiness and college grade points average, as well as indirect positive relationships with visualization, and OO processing through design readiness; and indirect negative relationships with web designing and college grade point average. For exam grades, analyses revealed positive direct relationships with visualization and college grade point average. And finally, overall grades for School A had a direct positively relationship to visualization, college grade point average, and design readiness, as well as indirect positive relationships with visualization and OO processing; and indirect negative relationships with web designing and college grade point average.

For School B, there were no significant relationships with lab grade. However, when examining the relationships with project grades, college grade point average was found to have a direct positive relationship and the cognitive measure of logical reasoning was found to have a direct negative relationship. Further, exam grades and overall grade had a direct positive relationship with college grade point average.

For the Combined Sample, lab grades were found to have direct positive relationships with college grade point average, web designing, and design readiness, as well as indirect positive relationships with school and visualization; various CS tasks was found to have a direct negative relationship to lab grades. It was also found that project grades had a direct positive relationship with school and college grade point average. Further, exam grades had direct positive relationships with school, college grade point average, and visualization. And finally, overall grades were found to have direct positive relationships with college grade point average and design readiness, as well as indirect positive relationships through the intermediate variable design readiness.

Research Question Three: Student Characteristics Related to Design Task

Not surprisingly, the pre-training design task score was positive and significantly associated with the post-training design task score across schools and combined sample. It was found that, for School A, the cognitive ability spatial orientation and the prior computer science experience computing platforms had a positive direct relationship with post-training design task score. For school B, only object-oriented processing had a direct positive relationship with post-training design task score. Analyses of the Combined Sample, revealed no direct or indirect relationships with design task. There were no negative relationships for either school or the combined sample.

Research Question Four: Design Readiness Related to OOD Course Performance

The analyses of the school-specific samples revealed that the design readiness assessment scale was significantly related to project grades and overall grades for School A. For School B, design readiness was not found to be significantly related to course performance. For the Combined Sample, lab grade and overall grade were associated with the design readiness assessment scale.

Research Question Five: Design Readiness Related to Design Task Performance

Analyses of the data revealed no significant direct or indirect relationship between the design readiness assessment scale and the post-training design task for either school or the combined sample.

CHAPTER FIVE

DISCUSSION

This exploratory study examined the relationships among (1) college grade point average; (2) cognitive abilities (spatial orientation, visualization, logical reasoning, flexibility, perceptual style); (3) prior computer science experience (UNIX programming, object-oriented processing, web designing, computing platforms, various CS); (4) design readiness (the design readiness assessment scale); and (5) OOD course performance (lab grade, project grade, exam grade, overall grade, post-training design task score).

Students (N = 161) enrolled in CS2 courses at two southeastern state institutions participated in this study during the spring semester, 2004 (January through May). Grades were obtained from the course instructors. The researcher sought to determine if the mean scores on the various assessments were significantly different across the schools. After evaluating the resulting data, it was found that the schools significantly differed on all measures of interest in this study, with the exception of college grade point average. The data were then analyzed based on School A, School B, and the Combined Sample.

The following five research questions were utilized to assess the relationships between the variables of interest in this study.

1. What student characteristics are related to design readiness?
2. What student characteristics are related to performance in an OOD course?
3. What student characteristics are related to performance on a design task?
4. Is design readiness related to a student's performance in an OOD course?
5. Is design readiness related to a student's performance on a design task?

The resulting findings will be assessed in the remainder of this chapter.

Research Question One: Student Characteristics Related to Design Readiness

At the time of this study, the principal investigator was unable to discover any supporting literature examining the antecedents of readiness. In recent years, however, studies examining the outcomes of readiness have reported that mathematical readiness is associated with overall academic performance (Heinze, Gregory, & Rivera, 2003; Jakiela & Fayad, 1986). Thus, this study was the first known attempt to identify antecedents of readiness—and particularly design readiness.

The path analytic results revealed that for School A, design readiness was positive and significantly associated with both cognitive ability visualization ($\beta = .29$) and the prior computer science experience object-oriented processing ($\beta = .40$). Conversely, design readiness was negatively and significantly associated with web designing ($\beta = -.45$) and college grade point average ($\beta = -.22$). For School B, design readiness was positive and significantly associated with visualization ($\beta = .26$). The results showed no other significant associations with design readiness. Additionally, for the Combined Sample, school and visualization were positive and significantly related to design readiness.

Plausible Explanation of the Results

This investigator determined that design readiness was positively and significantly associated with cognitive ability visualization for both School A, School B, and the Combined Sample. Those who scored higher on measures of visualization had a greater ability to mentally transpose or rotate objects from a simple one-dimensional shape to a three-dimensional shape. The association between visualization and design readiness was a significant finding. Specifically, this outcome suggested that cognitive measures of visualization can be linked to a general capacity to restructure/reorganize objects, where the design readiness assessment scale is a specific application of this general knowledge. The mental processing associated with solving a visualization problem instinctively correlates to the stages of the OOD problem-solving model. Both assessments require an individual to assess initial specifications and mentally restructure (decompose/recompose) them to fit into the provided solution restraints; guiding one from the stages of *understanding the problem* (OOD problem solving model, stage one: divide and conquer) to *carrying out the plan* (OOD problem solving model, Stage Four: abstraction/generality).

This study used the ETS Kit of Factor Referenced Cognitive Surface Development test to measure visualization. Despite the fact that a variety of instruments have been used to measure visualization, the findings have been generally consistent. Visualization was shown to be significantly associated with performance in introductory computer science courses, as well as in introductory engineering design courses (Scalan, 1988; Jakiela & Fayad, 1986).

While there was only one positive significant association found for design readiness within School B, the results of data analysis from School A revealed that object-oriented processing was also positively and significantly associated with design readiness. Analysis of the qualitative data found that 42% of students from School A reported having more than one year of work experience in a computer science related field, while only 21% of the students from School B reported similar levels of work experience. Jia (2003) found that the software industry has adopted object-oriented development paradigm as common practices. Therefore, one could safely assert that individuals with more work experience in a computer science related field are more likely to be familiar with object-oriented processing.

It was further noted that for subjects at School A, prior computer science experience involving Internet web design was negatively and significantly associated with design readiness.

In other words, individuals who reported higher levels of web application development were more likely to score lower on the design readiness assessment scale. When examining the qualitative data associated with the type of computing package used most often, desktop publishing was reported 66% more often in School A than in School B. Therefore, it is reasonable to assume that some students from School B would have developed web application development learning schemas. Piaget (1990) described schemas as the basic building blocks of knowledge and intellectual development. Schemas have further been described as "mental representations of general categories of objects, events, or people" (Chalmers, 2003, p. 597).Web application development and the design readiness scale may both require the user to rely on previously existing solutions, or libraries, to achieve the final product. However, web application development and paper-and-pencil design scenarios provide qualitatively different experiences (McDonald, 2002). This further affirms the finding of a significant negative association between design readiness and web designing.

Finally, it was observed that grade point average is negatively and significantly associated with the design readiness assessment scale—but only for School A. This negative finding, however, was not surprising. Independent-samples *t* test on demographics showed a significant difference in the age ($t(159) = 6.63$, $p = .00$), major ($t(159) = 6.89$, $p = .00$), and classification ($t(159) = 7.68$, $p = .01$) between samples. Typically, first-year college students within the same major are required to take similar courses. Beyond the first year of an academic program, the success measure of grade point average (GPA) is quite often susceptible to the stimuli of many other factors (i.e., varying courses, course difficulty, student study habits, etc.). Because of these influential factors, college GPA cannot be associated with a specific skill—and specifically should not be associated with OOD. These findings are consistent with Gall, Borg &

Gall (1996) who contended that GPA was a shifting, amorphous criterion, and therefore would be difficult to use as a predictor of a specific skill set.

It is quite noteworthy that the measure of cognitive ability – visualization – held a significant relationship with design readiness across school and combined sample. The measure of design readiness administered in the study – the Design Readiness Assessment Scale – reported internal consistencies ranging from .68 to .82. The administration of this instrument in the current study yielded internal consistencies between .32 and .54. While School A (α = .60) was above the acceptable standard, School B (α = .36) was well below the published minimum requirements. Kehoe (2000) reported that test reliability values as low as .50 were satisfactory for short tests (10 - 15 items). This researcher also stated that important decisions concerning individual students should not be based on a single test score when the reliability is less than .80. While the findings of the observed association between design readiness and visualization are promising, these results cannot be generalized beyond the current sample because of the less that desirable internal consistency of the design readiness assessment scale.

Research Question Two: Student Characteristics Related to Course Performance

The second point of inquiry for this research was to identify student characteristics that were associated with course performance. Lab, project, exam, and overall course grades were used as measures of course performance within this research. Even though overall course grade could be considered the pinnacle measure of course performance, it was thought that multiple measures of this variable would paint a richer portrait of course performance. And indeed, other researchers (e.g., Lending & Kruck, 2002; Evans & Simkins, 1989) have found that examining course performance measures separately provides more persuasive explanatory data.

Within this study there were several reasons for the use of multiple measures of OOD performance. To begin with, multiple measures were utilized because the particular course of interest in this study—CS2—involved more than OOD (i.e., programming projects, homework, and closed lab assignments). Furthermore, the overall course grade was confounded by multiple-instructor assessments. The lab sessions at both schools, for example, were taught by instructors other than the primary course lecturer. Because teaching effect was not a focus of this study, it was necessary to attempt to remove or isolate the effects of multiple instructors. Moreover, the assessment scheme across schools was slightly different. Thus, identifying measures that were common across different grading schemes could be considered a major contribution of this research.

The following subsections present a summary of the various findings across schools and combined sample, followed by plausible explanations of these findings.

Lab Grade

The lab grade was positively and significantly related to the prior computer science experience measure of object-oriented processing for School A. When examining School B, however, no significant relationships were found between lab grade and the measures of interest in this study. For the Combined Sample, it was found that college grade point average, web designing, and design readiness held positive direct relationships with lab grade, as well as school and college grade point average holding positive indirect relationships. It was also found that various CS, held a negative direct relationship with lab grade for the Combined Sample.

Project Grade

These results were consistent across both schools and combined sample, in that college grade point average was positively and significantly associated with project grade in School A and School B. In addition, for School A, the design readiness assessment scale was positive and directly related to project grade, as well as visualization and OO processing holding positive indirect relationships. Further, in addition to college grade point average, significant relationships for the Combined Sample revealed positive a direct relationship between project grade and school.

Exam Grade

The exam grade was positively and significantly related to college grade point average across both schools and the combined sample. While GPA was the only measure related to exam grade for School B, the cognitive measure of visualization was a direct positive relationship with exam grade for School A. Along with college grade point average, for the Combined Sample, visualization and school again held direct positive relationships with exam grade.

Overall Course Grade

The overall grade in the course revealed a direct positive relationship with college grade point average, the design readiness assessment scale, and the cognitive measure of visualization for School A, as well as positive direct relationships with visualization and OO processing. There were also negative indirect relationships between overall grade and web designing and college grade point average. Only college grade point average revealed a positive direct relationship with overall course grade for School B. For the Combined Sample, along with college grade point average, school and visualization held positive indirect relationships with overall grade.

Plausible Explanation of the Results

The results show that for School A, the cognitive measure visualization was positively and significantly associated with exam and overall grade. Furthermore, design readiness was positively and significantly associated with project grade and overall grade.

The underlying variable that was present across both school and performance measure was the use of college grade point average. Even though GPA was shown to be a poor predictor of a specific skill, its use in the general prediction of course performance or overall academic success is significant. As noted in Chapter 2, the literature has discussed varying models of academic performance. And while some consider GPA an amorphous variable, most researchers have agreed that one of the best predictors of future college GPA is current college GPA (Camara & Echternacht, 2000). Furthermore, numerous studies have found that college GPA is a viable predictor of individual class performance (e.g., Lending & Kruck, 2002; Camara & Echternacht, 2000; Chamillard & Karolick, 1999; Eskew & Faley, 1988).

The data for this study was analyzed with and without college GPA as a variable in the regression model. Without this measure in the regression models, it was found that prior computer science experience measures (object-oriented processing and computing platforms), as well as cognitive abilities (visualization and spatial orientation) were related to various measures of OOD course performance. The emergence of these four variables corroborated the underlying theme of *sophistication*. In other words, there is a certain level of mental sophistication that is necessary to be able to transition between varying platforms, design representations, or imagery. This research contended that this level of sophistication was best captured in terms of abstract versus concrete thinking. In *The Object-Oriented Thought Process*, Weisfield (2000) elucidated that students and software professionals who were able to understand design have developed the ability to view the world abstractly. Weisfield further believed that practical applications of this capability could be found in the way, or the level in which, one gave directions, performed simple tasks, or solved problems.

While it was hoped that visualization, spatial orientation, object-oriented processing, and computing platforms measures would be a robust set of associative measures of OOD course performance, the power of these variables were eclipsed by college GPA. Moreover, student characteristics associated with successful course performance were varied when GPA was added to the model. In fact, when college grade point average was introduced, it accounted for a large portion of the variance in the measures of OOD course performance measures. Although this was not the expected outcome for the present research, these results were consistent with other reports (Lending & Kruck, 2002; Evans & Simkins, 1989; Jakiela & Fayad, 1986). College grade point average actually provided a non-intrusive measure of course performance, unlike the resource expenditures (i.e., time, cost, mental capacity) required to obtain other measures.

Although the use of college grade point average offered more associative power than other variables within this study. This research does not ignore prevailing power of the cognitive measure of visualization above and beyond the generally strong relationship of college grade point average. The relationship found can be seen as a surrogate for IQ and motivation. Further, Visualization offered many interesting significant direct and indirect relationships through the mediated variable design readiness. Although the current measure of design readiness is only marginally reliable, the fact that visualization prevailed shows the strength of this measure of cognitive ability. This is encouraging findings for future work on the relationship between cognitive abilities and more reliable measures of design readiness.

It was also found that web designing was negatively related to overall grade for School A. A plausible explanation is that programming the web follows a paradigm that is very different from that taught in the CS2 course. Web designing required the use of several languages at the same time (client-server applications), and OOD decisions are taken in a very different context; for example the Tower of Hanoi problem requiring divide and conquer strategies. It is speculated that students introduced to procedural take a longer to transition to OO; speculations are the same for web programming. Therefore, when an instructor encounters students with specific programming paradigms other that OO, it is highly likely that those students will have a steeper OOD learning curve than those with experience in the OO paradigm. It may come to fact that the computer science programs may have to offer OOD later in the computer science curriculum and/or offer practical supplemental instruction on OOD principles/strategies.

Research Question Three: Student Characteristics Related to Design Task

The third area of inquiry examined the extent to which student characteristics were associated with post-training design task score. The design task was a specially constructed measure of object-oriented design performance. Because other OOD performance measures available for use in this study were confounded with more than OOD, there was a need to create a consistent measure that could be used across the schools. Thus, a design task was created to be fun, engaging, and at the appropriate level for novice designers. The literature supports the notion of identifying student characteristics associated with design task performance (Jakiela and Fayad, 1986). Jakiela and Fayad (1986) found that college grade point average was one of the strongest predictors of design performance. The present research assumed that college grade point average would emerge in a similar capacity.

After careful analysis of the data it was found that college grade point average was not significantly related to post-training design task score for either school. However, it was notable to find that spatial orientation and computing platforms held a direct positive relationship to post-training design task score for School A and object-oriented processing held a direct positive relationship to design task for School B. No measures of individual differences were found to hold a direct or indirect relationship to the post-training design task for the Combined Sample. Not surprisingly, it was also found that the pre-training design task score was positively and significantly associated with the post-training design task score across both schools and the combined sample.

Plausible Explanation of the Results

The finding that the cognitive ability spatial orientation is positive and significantly related to design task performance is exciting news for this research. The measure of spatial orientation used within this study – Cube Comparison – asked participants to mentally rotate a cube to determine if the second cube is a rotated version of the first cube. The design task may be seen as requiring the same mental rotation abilities. The design task requires the participant to be able to mentally rotate/transform the scenario/requirements, piece by piece, into specific classes; providing a direct mapping from the scenario to the design requirement. Scalan (1988) examined the use of spatial orientation and found a direct relationship between spatial orientation and programming ability. While this study is not interested in programming ability, it is encouraging to find that spatial orientation was found significant for School A. It was found that the demographic make-up of participants from School A was quite diverse and included a large portion (42%) of individuals with one or more years of programming work experience.

Object-oriented processing was shown to have a significant direct relationship to post-training design task for School B. This finding provides validation that the design task does indeed capture object-oriented skills/principles. The fact that no other measures were to be related to the design task, for School B, was surprising.

One plausible explanation for the lack of significance above and beyond object-oriented processing, for School B, was that the design task failed to capture *all* the phases of the design process as intended. However, this researcher maintains that design task may still prove to be an effective measure of OOD performance—and indeed, qualitative data has supported this belief. A number of students reported comments such as “This was fun!” “Wow, I learned a lot from this!” “Great Fun! I wonder how it would be to solve this in groups!” and “I wasn’t sure what I was doing when I complete this at the beginning of the semester, but I am excited that I understand it now!”

Another plausible explanation of the results was the fact that this task was completed external to the course. In addition, in order to maximize the design training timeframe, the post-training design task was given during the last week of the course. Not unexpectedly, a significant number of participants entered the design session looking lethargic and expressing varying levels of fatigue. For better or for worse, the underlying motivation to participate in this study was driven by the promise of receiving an ‘A’ on the final programming project in the course, regardless of the quality of their design. Lamentably, some students reported comments such as “Thanks for the A. Sorry this isn’t my best work.” “I didn’t really concentrate because I have to study for the final.” and “So do I really get an A, even if I put down anything?”

Although this research identified few measures associated with post-training design task score beyond the pre-training design task score, it was believed that the use of this design task as a graded course assignment may have provided different results.

Research Question Four: Design Readiness Related to Course Performance

The fourth point of inquiry for this research was the extent to which design readiness was associated with course performance. While prior research on the outcomes of design readiness could not be identified, the use of mathematical readiness techniques was prevalent in the literature (e.g., Heinze, Gregory, & Rivera, 2003; Mayer, Dyck, & Vilberg, 1986). These researchers reported that reliable measures of mathematical readiness were significantly associated with course performance, as well as with overall academic performance.

The current research revealed parallel conclusions for the design readiness assessment scale, despite its low reliability in the current study. It was found the design readiness assessment scale held a direct and positive relationship with overall course grade for School A and the Combined Sample. Again, it is important to highlight that the use of the design readiness assessment scale offered an indirect relationship with the cognitive ability visualization for School A and the Combined Sample. Despite the fact design readiness is not related to OOD course performance for School B, design readiness held a positive significant relationship when examining the Combined Sample. More reliable versions of the design readiness assessment scale are likely to produce even more positive relationships.

Plausible Explanation of the Results

It is likely that the design readiness assessment scale is more influential in School A because of the various backgrounds of the students. The students from School A were found to possess significantly less prior computer science experience than students from School B; however, the students from School A possessed higher levels of exposure to college curricula. Ninety –eight percent of the students from School B were classified as freshmen, where as classifications were distributed across classifications for School A. The results found can be speculated to be a measure of *maturation*. Existing research found that learning design is a measure of experience and exposure to complex design problems (Jia, 2003; Morelli, 2000;

Bucki & Stucki, 2001; Bergin, 1996; Coplien, 1996). The current research supports the research findings of Buck & Stucki (2001) that state the design too early can be harmful.

The design readiness assessment scale was found to be associated with overall course grade for School A and the Combined Sample. These results are promising in that the design readiness assessment scale encompasses six fundamental design principles covered over the entire semester. It was found that certain components of the lab grades, exam grades, and project grades assessed individual principles, where few covered the entire design process covered in the design readiness assessment scale. The overall course grade provides a culmination of the application and assessment of these strategies.

The fact that design readiness was not related to more measures of OOD course performance can be explained simply by reviewing the reliability of the design readiness assessment scale for the current study, namely .36 to .60. Although the pilot studies showed more promise, the current results showed that the design readiness scale failed to capture design constructs as illustrated in the OOD problem-solving model. Careful revisions of the design readiness scale will be performed before it could be used in supplementary studies.

Research Question Five: Design Readiness Related to Post-Training Design Task Score

The final point of inquiry for this research was the extent to which design readiness was related to the post-training design task score. This question was introduced into the study to assess *ability* versus *performance*. The literature has shown that there may be a significant gap between what one is capable of performing and what is actually performed. Wiggins (1992), for example, has argued that student performance is not only concerned with performing simplistic tasks that impart the desired bit of knowledge. Rather, it entails “putting it all together” with good judgment. The design readiness assessment scale and the design task were both created to measure ability. In fact, examining the relationship between these devices was considered pivotal to this research. As it turned out, however, the results showed that the design readiness scale was not significantly associated with the post-training design task score for either score or the combined sample.

Plausible Explanation of the Results

As previously stated, the design readiness assessment scale had marginal internal consistency in this study (Kehoe, 2000). Due to the novel instrumentation of both the design readiness scale and the design task, further research must be conducted using each scale separately so that they can be refined and studied again in a design training situation.

Limitations

While this study contributes to the literature with regard to measures associated with OOD performance, several methodological limitations in this study require additional discussion. For example, the measure of prior computer science experience was based solely on self-reported data, which is generally considered to be a fallible source because it relies on retrospective reporting and context-dependent assessments (Schwarz, 1999). Moreover, some participants could have been reticent or protective about the information that they were willing to report, while others may have had difficulty accurately recalling specific data (Gall, Borg, & Gall, 1996).

While many have used measures of self-reporting to capture prior computer science experience, the results of instrument success have been varied (e.g., Katz, Aronis, Allbriton, Wilson, & Soffa, 2003; Lending & Kruck, 2003; Rountree, Rountree, & Robin, 2002). Some recent studies involving prior computer science experience utilized instruments that required the participant to perform a specific task (Ventura, 2004; Wilson, 2000). As such, it would not be feasible to expect outcomes to be consistent across measures.

A second limitation of this study is the problem of mortality. This research experienced a 22% (n = 45) overall experimental mortality rate. The sample drawn from School A decreased by 25% (n = 25), with a loss of 20 men and 2 women. The sample drawn from School B decreased by 19%, with a loss of 15 men and 5 women. Disappointingly, the aggregate loss of women between the two schools represented a greater than 70% experimental mortality rate. It was impossible to control for this loss because a large portion (n = 40) withdrew from the course, and others expressed subject fatigue and opted not to continue participation.

Subject fatigue, testing, carry-over and floor effects are other limitations associated with this study. Because of the time-intensive data collection points outside of class, some students expressed discontent about participating in the study. Thus, there may have been participants that did not perform to the best of their abilities because of external conflicts—or merely participated for the benefit of opting out of the final course programming project. To control for the effects of subject fatigue, the researcher randomly ordered the administration of the outside of class instruments.

It should also be noted that testing effect was believed to have contributed to the lack of motivation in completing the final design session. Participants often would inquire about the amount of time that was required for the post-training design session, since the pre-training sessions took anywhere from 1.5 to 4 hours to complete.

Also, there was clearly a carry-over effect between the pre-training design and the post-training design task. Because of semester scheduling limitations, there was no way for the researcher to control for this effect.

The limitation of floor effect was observed when the researcher evaluated the change scores from the pre-training design task to the post-training design task. This effect could have been caused by the lack of motivation to complete the task, or by the fact that the task may have been too complicated for their understanding. This second suggested limitation, however, was not borne out by the pilot studies, which indicated that the task was at the appropriate level for novice designers.

Yet another limitation of this study was instrumentation. Because of the lack of prior research in the area of OOD prediction, this research design was exploratory in nature and relied on theory and previous research findings in the area of computer programming when selecting variables. The experimental design was not based on a specific academic prediction model, but rather on an accumulation of existing success stories, resulting in the use of over ten instruments.

It should be noted that some of the present study's instrument outcomes were not utilized in research because of the unreliability of the findings and/or errors in data collection procedures. For example, an online version of the Myers-Briggs Type Indicator (MBTI) was used in the study, which was administered to individuals before the pre-design task and the post-design task. Unfortunately, the observed MBTI profile obtained from the pre-test typically did not match the MBTI profile obtained at post-test. Thus, these findings proved to be inconclusive and the MBTI was eliminated as a measure in this study.

Finally, the sample was purposeful and non-random, which could have limited the validity of its external generalizability. Furthermore, external generalizability of the design readiness assessment scale was limited because of low internal consistency.

Despite the limitations discussed above, college grade point average remains a very credible and reliable predictor of successful course performance.

Implications

The purpose of this research was to (1) develop an OOD problem-solving model, outlining the progressive stages and principles needed for teaching and learning OOD; (2) construct and validate an instrument designed to measure object-oriented design readiness, namely, the Design Readiness Assessment Scale (DRAS); and (3) identify experiences and cognitive measures that could be associated with object-oriented design performance. These goals were developed in order to shed light on two poorly understood factors of student difficulty: (1) varying conceptualizations of the underlying principles/strategies of OOD, and (2) preparedness or readiness to learn OOD.

The OOD problem-solving model and the design readiness assessment scale clearly demonstrated potential of becoming viable assets to OOD pedagogy. While this research may have generated more questions than it answered, the findings described herein are nonetheless quite promising. For example, this study serves as a valuable baseline assessment measure of OOD problem solving, especially given the expectation that OOD preparedness/readiness research is likely to flourish in the near future.

The most exciting contribution of this research to the existing literature is the OOD problem-solving model. Linn and Clancy (1992, p. 511) stated that, "Instructors often assume that students can take their general problem-solving skills and discover specific software design skills on their own. Thus, students learn design skills through unguided discovery." There seems to be an assumption that if students learn to program by writing code of increasing size and complexity, they will then logically discover and apply the necessary design strategies in a successful manner. This sequential phenomenon, however, has not been reliably reported. In fact, students are failing to demonstrate an understanding of the connection between object-oriented programming structures (e.g., objects, classes, methods) and higher-level design strategies (e.g., abstraction, composition, design patterns) (Buck & Stucki, 2000; Kafura, 1998; Astrachan, 1996). The OOD problem-solving model created in this research can potentially guide students and instructors in OOD pedagogy.

There are two attributes of the OOD problem-solving model that can be helpful to instructors. First, the model clearly explicates what design *skills* a student will need to be able to solve a particular stage of a design problem. As noted earlier, the term "skills" refers to design strategies/principles. Second, the model is purposefully language-independent, allowing instructors to have freedom in design problem development.

Polya's mathematical problem-solving model is used extensively throughout literature as a foundation for technical problem solving models (Custer, Valesey, & Burke, 2001).The OOD problem-solving model was developed using the theoretical framework of Polya's mathematical problem solving model. Custer et. al (2001) used Polya's mathematical problem solving model to develop the Student Individualized Performance rubric. This rubric was used to assess the pre/post training design tasks. OOD pre-training ability (the DRAS) and the assessment of OOD performance (design task) were two distinct psychometric instruments used within this study – created using the same foundational model.

The design readiness assessment scale is a direct application of the OOD problem-solving model. Each stage in the model is represented by questions on the DRAS. To be candid, this investigator took a gigantic leap of faith when trying to create a measure of OOD readiness using real-world scenarios. Although the pilot testing of the DRAS showed it to be a reliable instrument within the confines of one university, its reliability fell nearly below acceptable standards when the sample was broadened.

To conclude, the results of this research *somewhat* supports the findings of Jakiela and Fayad (1986) in that college GPA was the most confounded variable used in this study and it emerged most often as the most significant predictor. In fact, one could consider college grade point average as the "catch-all" predictor of performance. However, for those with less college experience college grade point average was less influential. A noteworthy finding of this research was that college grade point average was the only consistent variable significantly associated with OOD course performance and design task performance. In the absence of college grade point average, the cognitive measure of visualization and object-oriented processing prior computer science experience were shown to be significantly associated with OOD course performance.

Efforts were made during this study to try and combine samples drawn from two local universities. Although the courses were similarly designed and the same textbook was used, there was simply no way to control for the varying teaching methodologies used at the two institutions. Moreover, it was clear once the data were collected that the universities differed on all measures of cognitive ability and prior computer science experience. Ultimately, this resulted in the use of two separate samples. Thus, when future studies are conducted, careful consideration should be taken and leveling measures should be implemented if samples are to be obtained from more than one institution.

This research was motivated by two factors (1) varying conceptualizations of the underlying principles/strategies of OOD, and (2) preparedness or readiness to learn OOD. So to answer the question – *what OOD principles/strategies should be taught* – this research stands by suggestions that (a) Divide & Conquer, (b) Interface, (c) Polymorphism/Inheritance, (d)

Encapsulation, (e) Information Hiding, (f) generality, and culminate learning of these strategies with the teaching of (g) Abstraction.

When examining the question – *when are student's ready to learn OOD* – this research offers a promising instrument that may identify those students that would (1) potentially succeed at OOD or (2) that may require supplemental instruction of OOD concepts. The DRAS ultimately promotes a more individualized-constructivist learning paradigm over the rote learning paradigms of the past.

The findings of this research suggest that prior OO experience is related to the successful performance in an OOD. College admissions can not require that students have a year of OO industry experience prior to admission into a computer science program; however, computer science programs should seriously investigate when OOD is actually taught in the curriculum. The results show that maturation in a computer science program may offer as much explanatory power as does industry experience. Therefore, being mindful of the internal consistency of the DRAS, this research loosely supports the model of School A. In that, there were students of varying experience levels and academic classifications in the course; offering a rich set of practical skills to draw upon.

A plausible speculation about students from School B is that they are still in the earlier stages of Bloom's taxonomy of learning OOD (Buck & Stucki, 2001). Students from School B may still be in the initial stages of learning, thought to resemble simple *mimicking* of the instructor; students from School A may have the practical experience necessary to conceptualize general OOD principles/strategies and apply them to more complex problems. Since students from School B generally completed only one semester of OO programming, it is likely that their mimicking is based on the limited exposure. The fact is that these students may not have the necessary level of *prior exposure* of complex design problems. The data suggests that School B potentially possess the cognitive ability to learn complex design concepts. An interesting future study would be to re-examine these students after they have further matriculated through the computer science program.

This research also speculates that design instruction based on or around the design task may offer a rewarding learning environment for students. Students reported that, although the task was long, it was engaging and required practical application of some topics that were discussed in class. Many stated that is was enjoyable to "put their learning to practice".

Future Research

The design readiness assessment scale failed to produce a significant path to *all* measures of course performance and design task, thus warranting further refinement of the design readiness assessment scale measurement. There are a number of refinement approaches that this research will consider with re-designing the DRAS. First of all, consideration should be given to the stage at which to begin refinement of the DRAS. One important question to be considered is, should refinement go back a far as the conceptual framework of the OOD problem-solving model? This research used Polya's mathematical problem solving model and the theoretical framework and the OOD problem-solving model and the DRAS. Since the work of Polya (1957), several researchers have identified possible OO and OOD problem-solving models (Guindon, Krasner, and Curtis, 1987; Kant, & Newell, 1984; Malhotra, Thomas, Carroll, and Miller, 1980; Brooks, 1977). Further research will examine the reliability to these models and possible frameworks for the DRAS.

Further, the DRAS used a single scenario for each questions. Future revisions may use generalized scenarios to cover multiple questions and design strategies. While this approach is promising, the complexity of the real-world design scenarios may be beyond the comprehension of novice designers.

Another consideration is the psychometric and/or evaluation properties of the DRAS. Future research will address the limitations of multiple choice versus an ordered-scaling of the responses. Using this approach, participants will be asked rank responses from "Most Correct" to "Less Correct" choices. This will address concerns of design trade-offs and multiple solutions to one design problem.

Future research will also examine the role of sophistication or *design maturity* in OOD course performance. While this research initially, sought to examine the pre-training design ability of freshman, it was found that the participants from School A were distributed across classifications. According to individual differences measures related to design readiness for each school, it may be that sophistication within the computer science program, rather than specific prior computer science experience, leads to one's preparedness or readiness to learn design.

In the research of Bucki and Stucki (2001), the question was asked if design too early in the computer science educational process is harmful. When examining data reported within this study, the researchers are inclined to answer yes to Buck and Stucki's question. However, the researcher is leery to provide a definitive answer based on the current reliability of the DRAS and the confoundedness of the course being used. Buck and Stucki showed how design could be *taught* using Bloom's Taxonomy of Learning; however, no consideration was given to the aptitude of pre-training ability of the student. A reliable measure of pre-training design readiness, coupled with effective teaching strategies, has the potential to create a workforce of talented designers, equipped with the fundamental skills to effectively undertake any system design problem.

A promising set of associative variables was identified during the exploration of various measures pertaining to course performance, such as visualization, spatial orientation, object-oriented processing, and computing platforms. This study identified these variables as central to the belief that continuing studies investigating abstract vs. concrete learning styles will provide a richer explanation of important student characteristics associated with OOD course performance. The use of measures such as Kolb's Learning Theory (Kolb, 1984), Gregorc's Mind-Styles (Gregorc, 1984), or a valid measure of the Myers-Briggs Type Indicator (Myers, McCaulley, Quenk, and Hammer, 1988) are likely to be highly elucidating, especially when combined with dependable data samples.

Future research will also examine the data to determine characteristics of those students that did not complete the study – either by dropping the course or not completing the post-training design task. It will be interesting to compare the characteristics of those that decided to complete the course against those that decided to drop the course. The analysis will explore the idea that there are predictive experiences that may explain why a student dropped the course.

And finally, some interesting results were found when comparing School A to School B. School A is a mid-sized comprehensive university; where as School B is a large research-intensive university. It was found that those that participated in this study, from School A had significantly lower high school grade point averages than those participants from School B. While School A performed significantly lower on all measures of cognitive ability, prior computer science experience (with the exception of UNIX programming), and pre-training design task score, School A significantly improved over School B on the post-training design task score. It was found that the schools did not significantly differ on the measure of final OOD course grade. It was also noted that the college grade point averages did not significantly differ across schools. It is interesting to find that although students entered the course with glaring dissimilarities, the outcome of the course was the same. Future research will specifically examine the effects of various schools (i.e. research-extensive, research intensive, comprehensive, community college) and program curriculum on course performance and attrition rates. The course syllabi can be found in Appendix B.

Although beyond the scope of the current research, endeavors will also be taken to identify the psychometric properties of college grade average. These properties may provide a generalized framework for the overall computer science and information technology programs.

This research does not solve all the teaching and learning problems of OOD. However, this research is a first step towards creating a standardized set of OOD principles. The current researchers are aware of the risks of trying to capture the understanding OOD principles in multiple choice or scalable items. OOD is an art not a science, but even artists begin with fundamental strokes and techniques that will later turn into a masterpiece.

We are simply trying to spur the academic and professional computer science industry to produce those simple strokes – core competency OOD principles. It is our ultimate goal to create a workforce of talented designers, equipped with the fundamental skills to effectively undertake any system design problem.

In summary, this research identified that instructors can not ignore individual differences when teaching OOD. The cognitive ability visualization, prior OO experience, and overall college grade point average should be considered when teaching OOD. As it stands, without identifying specific teaching strategies used at the schools within this study, this research implies that OOD may require a certain level of practical computer experience before OOD is introduced into the curriculum.

REFERENCES

ACM Computing Curricula. (2001). *Computing curricula 2001 final report*. Joint Task Force on Computing Curricula – IEEE and ACM.

Allen, B. (1992). *Cognitive differences in end user searching of a CD-ROM index*. Annual ACM Conference on Research and Development in Information Retrieval.

Alspaugh, C.A. (1972) Identification of some components of computer programming aptitude. *Journal of Research in Mathematics Education, 3*, 89-98.

Ancis, J. R. (1994). *Academic gender discrimination and women's behavioral agency self-efficacy*. Unpublished doctoral dissertation, University at Albany, New York.

Anderson, R.E. (1987). Females surpass males in computer science problem solving: Findings from the Minnesota computer literacy assessment. *Journal of Educational Computing Research, 3*(1), 39-51.

Andreae, P., Biddle, R., Dobbie, G., Gale, A., Miller, L. & Tempero, E. (2000). Experience teaching CS1 with Java, *Journal of Computer Science Education*, *14(*1-2).

Arnold, S., Bodoff, D., Coleman, H. Gilchrist, F. Hayes, *An Evaluation of Five Object-Oriented Development Methods*, Hewlett Packard Laboratories, Bristol, United Kingdom, Report No. HPL-91-52, June 1991.

Arsin Corporation (2001). *Internet Readiness: Building an Internet Ready Organization*. Arsin Corporation white paper.

Asch, S.H., & Witkin, H.A. (1948). Studies in space orientation. II. Perception of the upright with displaced visual fields and with body sloped, *Journal of Experimental Psychology*, *38*, 455-477.

Astrachan, O., Berry, G., Cox, L., & Mitchner, G. (1998). Design Patterns: An Essential Component of CS Curricula, *SIGCSE 1998*, Atlanta GA, 153-160.

Astrachan, O. (2001). OO Overkill: When Simple is Better than Not. *SIGCSE 2001*, Charlotte, NC.

Athman, C., Adam, R., & Turns, J. (2000). Using Multiple Methods to Evaluate a Freshman Design Course. *Frontiers in Education Conference*. Kansas City, MO.

Bagert, D. (1996). In Teaching the Object-Oriented Paradigm, Providing a Complete Picture is Essential. *Position paper for OOPSLA '96 workshop "Teaching and Learning Design in the First Academic Year"*, San Jose, CA.

Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Englewood Cliffs, NJ: Prentice Hall.

Bandura, A. (1997). *Self-Efficacy: The Exercise of Self-Control*. New York, NY: W.H. Freeman and Company.

Barkhi, R. (2002). Cognitive style may mitigate the impact of communication channel, *Information & Managemen*t, 39(8), 677-688.

Baudoin, C, & Hollowell, G. (1996). *Realizing the Object-Oriented Lifecycle*. Upper Saddle River, NJ: Prentice Hall.

Ball, S. (1977), *Motivation in Education*. Academic Press.

Berkley University. (2003). *The Berkley Math Readiness Project*. Berkley, CA.

Bergin, J. (1996). The object is computer science. In Karel (Ed.), draft of *A Gentle Introduction to the Art of Object Oriented Programming*. New York, NY: John Wiley & Sons.

Billings, R. S. and Wroten, S. P. (1978). Use of path analysis in industrial/organizational psychology: Criticisms and suggestions. *Journal of Applied Psychology, 63*, 677-688.

Bishop-Clark, C. (1998). An undergraduate course in Object-Oriented Software Design. *Proceedings from Frontiers in Education Conference '98*. Tempe, AZ.

Booch, G. (1994). O*bject-oriented analysis and design with applications* (2nd ed.). Benjamin/Cummings, Redwood City, CA.

Bong, M. (2001). Between- and within-domain relations of academic motivation among middle and high school students: Self-efficacy, task-value, and achievement goals. *Journal of Educational Psychology, 93*(1), 23-34.

Brooks, R. (1977). Towards a theory of the cognitive processes in computer programming, *International Journal of Man-Machine Studies*, 9, 737-751.

Bridle (2000). Astronomical Information Processing Center. Associated Universities Inc., Washington, D.C. http://aips2.nrao.edu/docs/html/design.html. Retrieved. March 16, 2004.

Buck, D. & Stucki, D. (2001). Design early considered harmful: graduated exposure to complexity and structure based on levels of cognitive development. *Proceedings from SIGCSE 2000*, Austin, TX, 75-79.

Bunderson, E.D., & Christensen, M.E. (1995). An analysis of retention problems for female students in university computer science programs. *Journal of Research on Computing in Education, 128*(1), 1-15.

Butcher, D. F., & Muth, W. A. (1989). Predicting performance in an introductory computer science course. *Communications of the ACM, 28*(3), 263-268.

Brush, S. G. (1991). Women in science and engineering. *American Scientist, 79*, 404-419.

Byrne, P., & Lyons, G. (2001). The effect of student attributes on success in programming. *Proceedings of the 6th annual conference on Innovation and technology in computer science education*, 49-52.

Camara, W. J., & Echternacht, G. (2000). *The SAT I and High School Grades: Utility in Predicting Success in College.* New York: College Entrance Examination Board.

Campbell, P, McCabe, G. (1984) Predicting the success of freshmen in a computer science major. *Communications of the ACM*. *27*(11), 1108 - 1113.

Capretz, L. (2002). *Human Factors in Software Engineering*, The Fifth International Conference on Humans and Computers, Tokyo, Japan.

Capretz, L. (2003). Personality types in software engineering, *International Journal of Human-Computer Studies*, 58(2). 207-214.

Carroll, J.B. (1974). *Psychometric tests as cognitive tasks: A new structure of intellect* (p. 74-16). Princeton, NJ: Educational Testing Service.

Cattell, R. B. (1971). *Abilities: Their structure, growth, and action.* Boston: Houghton Mifflin.

Cavaiani, T. P. (1989) Cognitive style and diagnostic skills of student programmers, *Journal of Research on Computing in Education, 21*(4), 411-420.

Chalmers, P. (2003). The role of cognitive theory in human–computer interface. *Computers in Human Behavi*or, 19(5).

Chamillard, A. & Karolick, D., (1999). Using learning style data in an introductory computer science course, *The Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education*, 291-295.

Chamorro-Premuzic, T. & Furnham, A. (2003). Personality traits and academic exam performance. *European Journal of Personality 17*, 237-250.

Choi, S. & Cairncross, S. (2001). Using interactive multimedia for teaching and learning object oriented software design. *ITiCSE '01*, Canterbury, UK.

Clancy, M. & Linn, M. (1999). Patterns and pedagogy. *SIGSCE '99*. New Orleans, LA.

Clay, M. (1992). *Becoming literate: The construction of inner control.* Portsmouth, NH: Heinemann.

Coplien, J. (1996). *Software Patterns*. SIGS Books and Multimedia, New York.

Cribbs, J. C. Roe & Moon, S. (1992). *An Evaluation of Object-Oriented Analysis and Design Methodologies*, SIGS Books, New York, New York.

Custer, R.L., Valesey, B.G., & Burke, B.N. (2001) An Assessment Model for Design Approach to Technological Problem Solving. Online Journal of Technology Education. Vol. 12-2.

Deckro, R.F. & Woundenberg, H.W. (1977). MBA admission criteria and academic success, *Decision Sciences*, 765-799.

Demetry, C. (2002). *Understanding Interactions between instructional design, student learning styles, and student motivation and achievement in an introductory materials science course*. 32nd ASEE/IEEE Frontiers in Education Conference.

Denelsky, G.Y. & McKee, M.G. (1974). Prediction of computer programmer training and job performance using the AABP test. *Personal Psychology,* 129-137.

Dewey, J. (1935). *Liberalism and Social Action*. New York: Putnam.

Dey, S., Mand, L. (1986) *Effects of mathematics preparation and prior language exposure on perceived performance in introductory computer science courses*. Proceedings of the Seventeenth SIGCSE Technical Symposium on Computer Science Education.

Dillon, W. R. & Goldstein, M. (1984). *Multivariate Analysis: Methods and Applications.* Wiley: New York.

Dijsktra, E.W. (1968). Go to statement considered harmful. *Communications of the ACM*, *11*(3), 147-149.

Donald, A. (2002). The impact of individual differences on the equivalence of computer-based and paper-and-pencil educational assessments. *Computer and Education*, 39(3).

Ekstrom, R. B., French, J. W., & Harman, H. H. (1979). Cognitive factors: Their identification and replication. *Multivariate Behavioral Research Monographs, 79*(2).

Entwisle, N. (1998). Motivation and approaches to learning: motivating and conceptions of teaching. In *Motivating Students,* Brown, S et al. (eds.), Kogan Page.

Eskew. R.K. & Faley, R. H. (1988). Some determinants of student performance in the first college-level financial accounting course. *The Accounting Review,* (LXII: 1).

Evans, G E. & Simkins, M G. (1989). What best predicts computer proficiency?, *Communications of the ACM, 32*(11), 1322-1327.

Fluery, A. (1999). Student conceptions of object-oriented programming in java. *Journal of Computing in Small Colleges 15*(1), 69-78.

Fichman, G. & Kemerer, C.F. (1991). *Object-oriented and conventional analysis and development methodologies: comparison and critique,* Center for Information Systems Research, Sloan School of Management, M.I.T., CISR WP.

Fowler, M. (1997). UML Distilled: *Applying the standard object modeling language*. Upper Saddle, NJ: Addison-Wesley.

Gall, M. D., Borg.W.R., & Gall, J.P. (1996). *Educational research: An introduction.* White Plains, NY: Longman.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J.(1995). *Design patterns: elements of reusable object-oriented software*. Addison Wesley, Reading, MA.

Garlan, D. (1996). *Software architecture perspectives on an emerging discipline*. Upper Saddle River, NJ: Prentice Hall, Inc.

Garton, B. L., Ball, A. L., & Dyer, J. E. (2002). The academic performance and retention of college agriculture students. *Journal of Agricultural Education*, *43*(1), 46-56.

Gibbs, D. (2000). *The effect of a constructivist learning environment for field-dependent/independent students on achievement in introductory computer programming*. 31st SIGCSE Technical Symposium on Computer Science Education, Austin, TX.

Gentili, K.L., McCauley, J.F., Christianson, R.K., Davis, D.C., Trevisan, M.S., Calkins, D.E., and Cook, M.D. (1999). Assessing students' design capabilities in an introductory design class. *Proceedings of Frontiers in Education Conference*, Puerto Rico, November.

Glorfeld, L. W., & Fowler, G. C. (1982). Validation of a model for predicting aptitude for introductory computing. Association for Computing Machinery Special Interest Group *Computer Science Education Bulletin, 14*(1), 140-143.

Goold, A., & Rimmer, R. (2000). Factors affecting performance in first-year computing, *ACM SIGCSE Bulletin, 32*(2), 39-43.

Goor, G. van den, Hong, S. and S. Brinkkemper, A. (1992). *Comparison of Six Object-oriented Analysis and Design Methods*. Report method Engineering Institute, University of Twente, The Netherlands, and Computer Information Systems Department, Georgia State University, Atlanta.

Green, S. B. & Salkind, N. J. (2003). *Using SPSS for Windows and Macintosh: Analyzing and understanding data*. Saddlebroook, NJ: Prentice Hall.

Greening, T. (1999). Gender stereotyping in a computer science course. *SIGCSE Bulletin, 31*(1), 203-207.

Gregorc, A. F. (1984). Style as a symptom: A phenomenological perspective. *Theory into Practice, 23*(1), 51-55.

Guindon, R., Krasner, H., and Curtis, B. (1987). Cognitive Processes in Software Design: Activities in Early, Upstream Design, in H.-J Bullinger and B. Shackel (eds.) INTERACT '87, 383-388.

Guzdial, M. (1995). *Centralized mindset: A student problem with object-oriented design and programming*. OOPSLA '97 Educator's Symposium.

Hahn, H., Hahn, J., & Kim, J. (1997). A cognitive engineering study on the development of an object-oriented process modeling formalism. *Proceedings of the Thirtieth Annual Hawaii International Conference on System Science*, 199-209.

Harrington, S.M. (1990). *Barriers to women in undergraduate computer science: The effects of the computer environment on the success and continuance of female students*. Unpublished doctoral dissertation, University of Oregon.

Harvard University. "What is Readiness?" An adaptation of Network readiness. http://cyber.law.harvard.edu/readinessguide/readiness.html. Retrieved April 24, 2004.

Heinze, L., Gregory, J, and Rivera, J. (2003). *Math readiness: the implications for engineering majors*. 33rd Conference on Frontiers in Education, Boulder, CO.

Hinkin, T. (1995). A review of the scale development practices in the study of organizations. *Journal of Management, 21,* 967-988.

Horstmann, C. (2004). *Object-oriented design & patterns*. San Jose, CA.: Wiley & Sons.

Jacobson, I., Booch, G. and Rumbaugh, J. (1999). *The unified software development process*. MA: Addison-Wesley.

Jakiela, M., & Fayad, L. (1989). *Identification of factors that contribute to engineering design skill*. Transactions of the IEEE.

Jia, X. (2003). *Object-oriented software development using java: principles, patterns, and frameworks*. MA: Addison Wesley Longman, Inc.

Jenkins, T. (2001). The motivation of students of programming. *Proceedings of the 6th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education ITiCSE 2001*, 53–56.

Jonassen, D. & Grabowski, B. (1993). *Handbook of individual differences, learning and instruction*. New Jersey: Lawrence Erlbaum Associates.

Jung, C.G. (1923). *Psychological Types*. London: Routledge & Kegan Paul.

Educational Testing Services (1976). *The Kit of Factor Referenced-Cognitive Tests – 1976 Edition*, Princeton, NJ.

Kafura, D. (1998). *Object-oriented software design and construction with C++*. Upper Saddle River, NJ: Prentice Hall.

Kant, E. and Newell, A. (1984). Problem Solving Techniques for the Design of Algorithms, *Information Processing & Management* 28, 97-118.

Katz, S., Aronis, J.D., Allbritton, C. Wilson, C & Soffa, M.L., (2003). An experiment to identify predictors of achievement in an introductory computer science course. *ACM Conference on Computer Personnel Research.*

Kehoe, J. (2000). *Basic item analysis for multiple-choice tests. practical assessment, research, and evaluation.* Retrieved from PAREonline.net.

Kolb, D. A. (1984). *Experiential learning: Experience as the source of learning and development*. New Jersey: Prentice Hall.

Kush, J. (1996). Field-dependence, cognitive ability and academic achievement in Anglo-American and Mexican-American students. *Journal of Cross-Cultural Psychology, 27*(5), 561-575.

Lending, D. & Kruck, S.E. (2002). What predicts student performance in the first college-level IS course? Is it different for men and women? *Proceedings of the 2002 ACM SIGCPR Conference,* May 14-16, 2002, Kristiansand, Norway, Munir Mandiwalla (Ed.), 100-102.

Lane, J., & Lane, A. M. (2001). Self-efficacy and academic performance. *Social Behavior and Personality*, 29, 687-694.

Lane, J., & Lane, A. M. (2002). Predictive validity of variables used to select students onto post-graduate courses. *Perceptual and Motor Skills.* 91, 649 -652.

Lee, R. & Tepfenhart, W. (1997). *UML and C++: A Practical Guide to Object-Oriented Development*. Upper Saddle River, NJ: Prentice Hall.

Leeper, R. (1990) A study of the personality types of successful computer science majors. *Proceedings from the ACM Conference on Computer Science,* 452.

Lewin, K. (1935). *A Dynamic Theory of Personality: Selected Papers.* N.Y.: McGraw-Hill.

Lewis, T., Perez-Quinones, M., & Rosson, M.B. (2004). *A measure of design readiness.* Proceedings from the 34th Conference on Frontiers in Education, Savannah, GA.

Linn, M.C., Clancy, M.J. (1992). Can expert's explanations help students develop program design skills? *International Journal of Man-Machine Stu*dies. 36(4), 511-551.

Liu, M. & Blanc, L. (1996). *On the retention of female computer science students.* Presented at the ACM SIGSE Conference, Philadelphia, Pennsylvania, February.

Lutz, M. (1999). *Using patterns to teach software subsystem design*. 29th ASEE/IEEE Frontiers in Education Conference, San Juan, Puerto Rico, session 11b4-21.

Maciel, R.G., Fernandez A., & Garrido, A. (1996). *A design toolbox for first academic year students.* Position paper for OOPSLA '96 workshop: Teaching and Learning Object Design in the First Academic Year.

Malhotra, A., Thomas, J.C., Carroll, J.M., and Miller, L.A. (1980) Cognitive processes in design, *International Journal of Man-Machine Studies*, 12, 119-140.

Matthews, D., Klaassens, A., Walter, L. and Stewart, T. (1999). LinguaLinks Library, Version 4.0, published on CD-ROM by SIL International, http://www.sil.org/lingualinks/literacy/ReferenceMaterials/GlossaryOfLiteracyTerms/ WhatIsReadingReadiness.htm#context.

Margolis, J. and Fisher, A. (2002). *Unlocking the Clubhouse: Women in Computing*. Cambridge, MA: MIT Press.

Martin, Robert (1996). *An introduction of object-oriented design*, retrieved on March 1, 2003 from http://www.accu.org/acornsig/public/articles/ood_intro.html

Mayer, R. E., Dyck, J. L., & Vilberg, W. (1986) Learning to program and learning to think: what's the connection? *Communications of the ACM, 29*(7), 605-610.

Mazlack, L J. (1980). Identifying potential to acquire programming skill, *Communications of the ACM, 23*(1), 14-17.

McClelland, D. (1961). *The Achieving Society,* Princeton , New Jersey: Van Nostrand.

McCracken, M. (2002). *Models of designing: understanding software engineering education from the bottom up*. Proceedings of the 15th Conference on Software Engineering Education and Training (CSEET '02).

Liu, M.L. & Blanc, L. (1996). On the retention of female computer science students, *ACM SIGCSE Bulletin, 28*(1), 32-36.

Monarchi, D.E. & Puhr, G.I. (1992). A research typology for object-oriented analysis and design, *Communications of the ACM*, *35*(9), 35-47.

Morahan-Martin,M.A., Olinsky, J.N. & Schumacher, P. (1992). *Gender Differences in Computer Experiences, Skills and Attitudes Among Incoming College Students*, Collegiate Microcomputer.

Morelli, R. (2002). *Object-Oriented Problem Solving*. Upper Saddle River, NJ: Prentice Hall.

Morrison, M. & Newman, T. S. (2001). A study of the impact of student background and preparedness on outcomes in CS I, *Proceedings from SIGCSE 2001*, Charlotte, NC,179-183.

Multon, K. D., Brown, S. D., & Lent, R. W. (1991). Relation of self-efficacy beliefs to academic outcomes: A meta-analytic investigation. *Journal of Counseling Psychology, 38*, 30-38.

Myers, I. B., McCaulley M. H., Quenk N. L. & Hammer A. L. (1998). *Manual: A guide to the development and use of the Myers-Briggs type indicator*. Consulting Psychologists Press, Palo Alto (CA).

Northrop, L.M. (1993). Finding an educational perspective for object-oriented development. *Computer Science Education, 4*(1), 5-12.

Nunnally, J. C. (1978). *Psychometric theory* (2nd ed.). New York: McGraw-Hill.

The Object Agency. (1996). A comparison of object-oriented development methodologies. Retrieved on March 16, 2004 from http://www.toa.com/smnn?mcr.html#se.

OOPSLA. (2000). Early Object Oriented Design Education Workshop, organized by Robert Biddle, Rick Mercer, Eugene Wallingford, retrieved January 17, 2002 from http://www.mcs.vuw.ac.nz/research/design1/.

Onish, L., & Duendes, D.S. (2000). *Math readiness: kindergarten*. Scholastic Publishing.

Orsak, G.C., & Acosta, M. (2002). *Proceedings from the Math and Science Readiness Institute*, The Institute for Engineering Education at SMU.

Pajares, F. (2002). *Overview of social cognitive theory and of self-efficacy*. Retrieved April, 10, 2004, from http://www.emory.edu/EDUCATION/mfp/eff.html.

Parnas, D.L. (1972b). On the criteria to be used in decomposing systems into modules, *CACM 15*, 1052-1058.

Pearl, A., Pollack, M., Riskin, E., Thomas B., Wolk, E. & Wu. A. (1990). Becoming a computer scientist: A report by the ACM committee on the status of women in computing. *Communications of the ACM, 33*(11), 47-58.

Pedhazur, E.J. (1982). *Multiple regression in behavioral research: Explanation and prediction* (2nd ed.). New York: Holt, Rinehart and Winston.

Petersen, C.C., & Howe, T.G. (1979). *Predicting academic success in introduction to computers*. Association of Educational Data Systems Journal, 182-191.

Peterson, R.A. (1994). A meta-analysis of Cronbach's coefficient alpha. *Journal of Consumer Research 21*, 381-391.

Polya, G. (1957). *How to solve it.* Garden City, NY: Doubleday and Co., Inc.

Proulx, V. (2000). Programming patterns and design patterns in the introductory computer science course. *SIGCSE '00*. Austin, TX.

Provost, J. A., & Anchors, S. (1987). *Applications of the Myers-Briggs type indicator in higher education.* Palo Alto, CA: Consulting Psychologists Press.

Rumbaugh, J. (1996) *OMT Insights*. SIG Books, New York, NY.

Rappin, N. (1997). S*tudents' psychological type and success in different engineering programs.* 27th ASEE/IEEE Frontiers in Education Conference.

Rappin, N. (1997). *A framework for teaching learners to model by focusing complexity of modeling and simulation tools*. Dissertation: Georgia Tech, College of Computing.

Reif, H. L. & S. E. Kruck. (2001). Integrating student groupwork ratings into student course grades. *Journal of Information Systems Education*, 12(2).

Rist, R. (1989). Schema Creation in Programming, *Cognitive Science, 13*, 389-414.

Custer, R.L. Valesey, B.G., & Burke, B.N. (2001). An assessment model for design approach to technological problem solving. The *Journal of Technology Education, 12*(2).

Rosati, P. (1998). *The learning preferences of engineering students from two perspectives*. 28th ASEE/IEEE Frontiers in Education Conference.

Rosati, P. (1999). *Specific differences and similarities in the learning preferences of engineering students*. 29th ASEE/IEEE Frontiers in Education Conference, #1544.

Rosson, M.B., Shaw, M., and Alpert, S.R. (1990). The Cognitive Consequences of Object-Oriented Design. *Human Computer Interaction 5*. Lawrence Erlbaum Associates, 345-379.

Rosson, M.B. & Carroll, J.M. (1997). Expertise and instruction in software development. In M. Helander & T.K. Landauer (Eds.), *Handbook of Human-Computer Interaction*, Second Edition. Amsterdam: North Holland, 1105-1126.

Rountree, N., Rountree, J. & Robins, A.V. (2002). Predictors of success and failure in a CS1 course. *Special Interest Group on Computer Science Education Bulletin,* 34(4):121-124.

Rumbaugh, J. (1996). *OMT Insights*. SIG Books.

Ryan, C. & Al-Qaimari, G.A. (1996). Cognitive Perspective on Teaching Object-Oriented Analysis and Design. In *Proceedings of the Annual Computer Science Postgraduate*

Conference, Technical Report (TR-96-36), RMIT University, Melbourne, Australia, Obtained: http://citeseer.nj.nec.com/13036.html

Sackrowitz, G.M., & Parelius, A. P. (1996). *Women in the introductory computer science courses.* Proceedings of 27th SIGCSE Technical Symposium on Computer Science Education.

Sandler, B.R. (1987). The classroom climate: Still a chilly one for women. In C. Lasser (Ed.), *Educating men and women together: Co-education in a changing world*, 113-123. Urbana: University of Illinois Press.

Scanlan, D. (1988). *The Mental Abilities Associated with Programming Aptitude*, IEEE Software.

Schwarz, N. (1999). Self-reports: How the questions shape the answers. *American Psychologist, 54*, 93-105.

Scragg, G. and Smith, J. (1998). *A study of barriers to women in undergraduate computer science*. Proceedings of the 29th SIGCSE Technical Symposium on Computer Science Education, Atlanta, GA.

Shade. B. (1981). Racial variation in perceptual differentiation. *Perceptual and Motor Skills, 52*(1), 243-248.

Sheetz, S.D., Puhr, G.I., Nelson, H.J. & Monarchi, D.E. (1995). *Student views of the difficulties of using object-oriented techniques*. Proceedings of PRISM.

Sidbury, J. (1986) A statistical analysis of the effect of discrete mathematics on the performance of computer science majors in beginning computing classes, *17th SIGCSE Technical Symposium on Computer Science Education,* 134-137.

Software Engineering Institute, Carnegie Mellon University, 2004, retrieved April 15, 2004 from http://www.sei.cmu.edu/str/descriptions/oodesign_body.html.

Sommerville, I. (2001). *Software Engineering*. Upper Saddle, NJ: Addison-Wesley.

Stevens, L.J., Wileman, S., & Konvalina, J. (1981). *Group differences in computer aptitude*. Association of Educational Data Systems Journal, 84-95.

Taylor, H. G. & Mounfield, L., C. (1994). Exploration of the relationship between prior computing experience and gender on success in college computer science. *Journal of Educational Computing Research, 11*(4), 291-306.

Teague, J. (1998). Personality type, career preference and implications for computer science recruitment and teaching, *Proceedings of the Third Australasian Conference on Computer Science Education*, 155-163.

Tegarden, D.P. & Sheetz, S.D. (2001). Cognitive Activities in OO Development, *International Journal of Human-Computer Studies, 54*(6), 779-798.

Ventura, P. (2004). Unpublished Dissertation: *On the origins of programmers: Identifying predictors of success for an objects-first CS1*. Computer Science, University at Buffalo, SUNY.

W-3C: World Wide Web Consortium. Object-Oriented Programming. Retrieved on April 15, 2004 from http://www.w3c.org.

Wallingford, E. (1996a). Toward a first course on object-oriented patterns. *SIGCSE*, Philadelphia, PA.

Wallingford, E. (1996b). Teaching Object Patterns in the First Course. *Position paper for OOPSLA '96, Teaching and Learning Object Design in the First Academic Year.* San Jose, CA.

Weisfeld, M. (2000). *The object-oriented thought process.* Indianapolis: Sams Publishing.

Weiss, D. (1976). Multivariate procedures. In Dunnette, M.D. (Ed.), *Handbook of industrial/organizational psychology*. Chicago, IL: Rand McNally.

Werth, L. (1989). *Predicting student performance in a beginning computer science class,* Proceedings of the 17th SIGCSE Technical Symposium on Computer Science Education, Cincinnati, Ohio.

Wiggins, G. (1992). Creating tests worth taking, *Educational Leadership*, 49, 26-33.

Wilson, B. (2000). *Contributing factors to success in computer science: A study of gender differences*. Unpublished dissertation from the Department of Curriculum and Instruction, Southern Illinois University at Carbondale.

Wilson, B. C., & Shrock, S. (2001). Contributing to success in an introductory computer science course: a study of twelve factors, *Proceedings of the Thirty-Second SIGCSE Technical Symposium on Computer Science Education*, 184-188.

Wileman, S.A., Konvalina, J. & Stephens, L.J. (1981). Factors influencing success in beginning computer science courses, *Journal of Educational Research*, *74*(4), 223-226.

Witkin, H., Moore, C., Goodenough, D., & Cox, P. (1977). Field-dependent and field-independent cognitive styles and their educational implications. *Review of Educational Research, 47*, 1-64.

Witkin, H., Oltman, P., Raskin, E., & Karp, S. (1971). *A manual for the embedded figures test*. Palo Alto, CA: Consulting Psychologists Press.

Wirfs-Brock, R., Wilkerson, B., & Wiener, L. (1990). *Designing object-oriented software*. Englewood Cliffs, NJ: Prentice Hall.

Wirfs-Brock, R., McKean, A. (2003). *Object design – roles, responsibilities, and collaborators.* Boston, MA. Addison-Wesley.

APPENDIX A

INFORMED CONSENT FORM

IRB Proposal for

Research Study of
A Measure of Design Readiness: The Effects of Individual Differences on Learning Object-Oriented Design

Submitted by
Tracy L. Lewis, Ph.D. Candidate
Manuel Pérez-Quiñones, Professor (Co-Advisor)
Mary Beth Rosson, Professor – Penn State (Co-Advisor)

Department of Computer Science
Virginia Polytechnic Institute and State University

Request for Exemption of Research Involving Human Subjects

[please print or type responses below]

Investigators(s): **Tracy L. Lewis, Manuel Pérez-Quiñones, Mary Beth Rosson**

Department(s): **Computer Science** Mail Code: **0106** E-mail: **{tracyL, perez}@vt.edu, mrosson@psu.edu**

Project Title: **A Measure of Design Readiness: The Effects of Individual Differences on Learning Object-Oriented Design**
\# of Human Subjects: **650**

Source of Funding Support: __**X**__ Departmental Research ____ Sponsored Research (OSP No.:______________)

[ ] All investigators of this project are qualified through completion of the formal training program or web-based training programs provided by the Virginia Tech Office of Research Compliance.

***Note*:** To qualify for Exemption, the research must be (a) of minimal risk to the subjects, (b) must not involve any of the special classes of subjects, and (c) must be in one or more of the following categories. A full description of these categories may be found in the Exempt Research section of the Virginia Tech "*IRB Protocol Submission Instructions Document* or in the federal regulations [45 CFR 46.101(b)(1-6)].
(http://ohrp.osophs.dhhs.gov/humansubjects/guidance/45cfr46.htm#46.101)

Please mark/check the appropriate category or categories below which qualify the proposed project for exemption:

[X] 1. Research will be conducted in established or commonly accepted educational settings, involving normal educational practices [see item (1), page 6 of the "Instructions" document].

[X] 2. Research will involve the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures or observation of public behavior, **unless** the subjects can be identified directly or through identifiers linked to the subjects **and** disclosure of responses could reasonably place the subjects at risk or criminal or civil liability or be damaging to the subjects' financial standing , employability or reputation [see item (2), page 6 – "Instructions"].

[ ] 3. Research will involve the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures, or observation of public behavior that is not exempt under item 2) above **if** the subjects are elected or appointed public officials or candidates for public office; **or** Federal statute(s) require(s) that the confidentiality or other personally identifiable information will be maintained [see item (3), page 6 of the "Instructions" document].

[ ] 4. Research will involve the collection or study of existing data, documents, records, pathological specimens, or diagnostic specimens if these sources are publicly available or if the information is recorded by the investigator in such a manner that subjects cannot be identified directly or through identifiers linked to the subjects [see item (4), page 7 of the "Instructions" document].

[ ] 5. Research and demonstration projects designed to study, evaluate, or otherwise examine public benefit or service programs, procedures for obtaining benefits or proposed changes in such programs [see item (5), page 7 of the "Instructions" document].

[ ] 6. Taste and food quality evaluation and consumer acceptance studies [see item (6), page 7-"Instructions].

Investigator(s)	Print name: **Tracy L. Lewis**	Date
Investigator(s)	Print name: **Manuel Pérez-Quiñones**	Date
Departmental Reviewer	Print name: **Dennis Kafura**	Date
Chair, Institutional Review Board		Date

rev. 4/27/01

Justification of Project

This proposed research will evaluate effective measures of assessing a student's ability to learn object-oriented design (*design readiness*). In particular, we will examine the effectiveness, reliability, and validity of a design readiness assessment scale aimed to measure the design aptitude of undergraduate computer science students.

Measuring one's ability to comprehend and effectively use object-oriented design (OOD) concepts is a complex task in the field of computer science. Although most computer science students can be *taught* to use OOD concepts, very few have mastered these techniques in such a way that would allow them to reuse the techniques in other areas of academic studies. The fundamental problem lies in one's concrete vs. abstract perception abilities. The goal of this study is to identify those areas of individual differences - gender, classification, experience, learning style, design readiness level - that can be utilized by OOD professors in developing robust teaching material and curricula aids.

Data collected by the proposed project will be used by Tracy L. Lewis in support of her dissertation in the department of Computer Science.

Procedures

This study will be executed throughout the 2003-2004 academic year, encompassing approximately 650 students enrolled in two new computer science courses - *CS 1705: Introduction to Object-Oriented Programming I* and *CS 1706: Introduction of Object-Oriented Programming II*. The courses will be taught by professors Steven Edwards (CS 1705), Dwight Barnett (CS1705), and Manuel Pérez-Quiñones (CS1706), and assisted by Tracy L. Lewis (Ph.D. Candidate and Graduate Teaching Assistant). Steven Edwards and Manuel Pérez-Quiñones have been consulted in the design and implementation of the requirements of this study as related to course administration.

Participation in this study involves the following:

- Initial demographic assessment (performed by the entire class)
- Design readiness assessment scale
- Group Embedded Figures Test
- Mid-semester Assessment
- Informal interviews/discussions
- End of year assessment (performed by the entire class)

Selection of Participants

As a part of regular course instruction and assignments, all students enrolled in CS1705 and CS1706 are required to complete the initial demographics assessment. Currently CS1705 has 3 lecture sections and 8 closed-lab sections and CS1706 has 2 lectures and 4 closed lab sections. Both have will have approximately 130 students per lecture section. Professors and teaching assistants will use the data from this assessment to partner students for the closed lab assignments.

Design Readiness Assessment Scale Pilots

Participants in the evaluation of the design readiness assessment scale will be drawn from all sections of CS1705: Introduction to Programming. It is anticipated that approximately 80 students will volunteer to participate in this study. All participants will complete the Group Embedded Figures Test to measure field dependence and independence.

Group Embedded Figures Test (GEFT)

This is a commercially available test timed test that will take approximately 20 minutes and requires participants to find common geometric shapes in a larger design—this simple assessment yields a wealth of information about field dependence-independence. The Group Embedded Figures Test (GEFT) was developed for research into cognitive functioning, but it has become a recognized tool for exploring analytical ability, social behavior, body concept, preferred defense mechanism and problem solving style as well as other areas. The GEFT is a 25 item assessment contained in a 32 page non-reusable booklet. We are in the process of purchasing the test now.

Pilot I

The purpose of pilot I is to evaluate item question wording, participant ranking of level of difficulty, participant design rationale, and generate new items. Approximately 60 students in three groups of 20 will participate in this study. Group 1 will be provided with a multiple choice design readiness scale and asked to answer each question to the best of their ability and provide insight/rationale for their decision. Group 2 will be provided with a multiple choice design readiness scale that has the correct solution supplied, and they will be asked to provide rationale as to why that solution is the most appropriate. Group 3 will be provided with a multiple choice design readiness scale that has design rationale supplied for each question, and they will be asked to select the best answer for each question based on the provided rationale. After completing the entire question set, all participants will then be asked sort questions into three categories of ascending difficulty, "basic", "intermediate", and "advanced". They will be instructed that not all categories have to contain items, and the number of items per category did not have to be equal. Following this, participants will be asked to identify possibly "confusing" questions.

Pilot II

The purpose of pilot to is to evaluate the construct validity and internal reliability of the design readiness assessment scale. Approximately 20 students from CS1705 will participate in this portion of the research. Participants will be provided with a multiple choice design readiness scale and asked to answer each question to the best of their ability and provide insight/rationale for their decision.

Design Readiness Assessment Scale Reliability & Validity Experiment

The purpose of this experiment is to test the design readiness assessment scale for reliability and validity on a representative sample of students in CS1706: Introduction to Object-Oriented Design. We are expecting approximately 60 participants in this experiment. We are accepting only those students that did not participate in the pilot studies. Participants will be provided with a multiple choice design readiness scale and asked to answer each question to the best of their ability.

Collection and use of self-assessment data

Several assessment measures will be used through this investigation.

Initial Demographics Survey

During the first week of courses all students enrolled in CS1705 and CS1706 will complete an online survey to obtain various demographic data – programming experience, general computer skills/proficiency information, general object-oriented programming and design questions, and computer use confidence. Students will be instructed that answering questions incorrectly will not have an adverse effect on his/her grade. Completion of the survey counts for one homework point, regardless of the correctness of their answers. These data will be used in aggregate form as a part of Tracy L. Lewis' dissertation as well as provide a mechanism for professors and teaching assistants to partner the students in the closed lab activities.

Mid-Semester Cognitive Assessment

Approximately halfway into the course – just before midterms – we will conduct a short survey of those students that volunteered to participate in this experiment. The survey will assess their perception of their current status in the course and their understanding of various object-oriented programming and design concepts.

Data collected by this portion of the project will be kept confidential by the investigators and will only be stared with the course instructor in an anonymous fashion, protecting the student's anonymity. The assessments (both demographic and mid-year) will be correlated with other student-specific data collected in this investigation, but the resulting data will be anonymous. The assessment data will be destroyed upon completion of this dissertation.

End of year assessment

During the final week of courses, all students enrolled in CS1705 and CS1706 will be asked to complete an end of the year self-assessment survey on their knowledge of object-oriented programming and design. Students will be instructed that this survey will count as one homework point, regardless of correctness of answers to questions requiring knowledge of object-oriented programming and design.

Collection and use of academic grades

For the participants in this study, all academic work completed for CS1705 and CS1706 will be collected and evaluated by the investigators during this study. This includes (but not limited to):

- Graded results from homework and quizzes
- Graded exams
- Auto-graded results from take-home projects
- Hand-graded results from take-home projects
- Auto-graded results from closed lab assignments
- Overall course grades

All coursework collected by this portion of the investigation will be stored securely (either electronically or physically) and viewable only by the course instructor, additional teaching assistants (as needed to perform their traditional activities), the project investigators, or the student who complete the work. Upon completion of the dissertation writing process, all data will be returned to the course instructor for disposal or further storage, as per department/university policy.

Collection and use of anecdotal data
Anecdotal data may also be collected during this investigation. This will come in the form of e-mails (both solicited and unsolicited) submitted to the investigators by the students, informal verbal anecdotes related to the study (in and out of the classroom), and transcriptions of group electronic communication mediums employed normally in the administration of the course (LISTSERVs and web-based message boards). All these data will be destroyed upon completion of the dissertation writing process.

Payment for participation
Participation in this study beyond the pre/post survey (these are required by the professor for all students enrolled in the course) may result in a small monetary stipend or free pizza and soda, consistent for all participants pending funding.

Risks
There are less than minimal risks to the study participants. The involvement and performance of students in this research is strictly voluntary. A small number of participants may experience eye strain from using a computer screen, or uncomfortable feelings from being watched or interviewed about their experiences in the courses.

Benefits
While there are no direct benefits to the participants from this research (other than payment for completion of assessment scales and survey), the participant will find the results of this study will further prepare professors on appropriate methods for teaching object-oriented design to individuals with vary levels of design aptitude.

Confidentiality and Anonymity
No one other than investigators will have access to assessment scale results and anecdotal data; both investigators and professors will have access to pre/post survey results. Anyone requesting this data will have access without express written consent from study participants. Likewise, no printed or electronic rendition of the information that could be directly attributed to a participant will be available to anyone other that the investigators without additional written consent from the participant. Otherwise, and presentation of this research will replace participant names, with anonymous codes or names and/or will report data in summarized form only. Any visual data included in professional presentations and publications will be used anonymously. No information identifying participants will accompany visual material.

Informed Consent
Enclosed.

VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY
Informed Consent for Participants in Research Projects Involving Human Subjects

Title of Project: A Measure of Design Readiness: The Effects of Individual Differences on Learning Object-Oriented Design

Investigator(s): Tracy L. Lewis Ph.D. Candidate; Dr. Manuel Pérez-Quiñones, Professor; Dr. Mary Beth Rosson, Professor - Penn State

I. Purpose of this Research/Project

You are invited to participate in a research project that evaluates effective measures of assessing a student's ability to learn object-oriented design (*design readiness*). In particular, we will examine the effectiveness, reliability, and validity of a design readiness assessment scale aimed to measure the design aptitude of undergraduate computer science students.

Measuring one's ability to comprehend and effectively use object-oriented design (OOD) concepts is a complex task in the field of computer science. The fundamental problem lies in one's concrete vs. abstract perception abilities. The goal of this study is to identify those areas of individual differences - gender, classification, experience, learning style, design readiness level - that can be utilized by OOD professors in developing robust teaching material and curricula aids.

This research project will solicit participation of students from CS1705: Introduction to Programming and CS1706: Introduction to Object-Oriented Design.

II. Procedures

Participants involved in the research will be recruited by in-class announcements / classroom invitations and over course LISTSERVs. If you agree to participate you could be asked to take part in the following research activities:

- ***Initial Demographics Survey***
  During the first week of courses all students enrolled in CS1705 and CS1706 will complete an online survey to obtain various demographic data – programming experience, general computer skills/proficiency information, general object-oriented programming and design questions, and computer use confidence. Students will be instructed that answering questions incorrectly will not have an adverse effect on his/her grade. Completion of the survey counts for one homework point, regardless of the correctness of their answers. These data will be used in aggregate form as a part of Tracy L. Lewis' dissertation as well as provide a mechanism for professors and teaching assistants to establish partners in the closed lab activities.
- ***Group Embedded Figures Test (GEFT)***
  This is a commercially available test timed test that will take approximately 20 minutes and requires participants to find common geometric shapes in a larger design—this simple assessment yields a wealth of information about field dependence-independence. The Group Embedded Figures Test (GEFT) was developed for research into cognitive functioning, but it has become a recognized tool for exploring analytical ability, social behavior, body concept, preferred defense mechanism and problem solving style as well as other areas. The GEFT is a 25 item assessment contained in a 32 page non-reusable booklet.
- ***Design Readiness Assessment Scale***
  Participants will be provided with a multiple choice design readiness scale and asked to answer each question to the best of their ability.
- ***Mid-Semester Cognitive Assessment***
  Approximately half into the course – just before midterms – we will conduct a short survey of those students that volunteered to participate in this experiment. The survey will ask assess their perception of their current status in the course and their understanding of various object-oriented programming and design concepts.
  Data collected by this portion of the project will be kept confidential by the investigators and will only be shared with the course instructor in an anonymous fashion, protecting the student's anonymity. The assessments (both demographic and mid-year) will be correlated with other student-specific data collected in this investigation, but the resulting data will be anonymous. The assessment data will be destroyed upon completion of this dissertation.

- ***End of year assessment***
  During the final week of courses, all students enrolled in CS1705 and CS1706 will be asked to complete an end of the year self-assessment survey on their knowledge of object-oriented programming and design. Students will be instructed that this survey will count as one homework point, regardless of correctness of answers to questions requiring knowledge of object-oriented programming and design.
- ***Collection and use of academic grades***
  For the participants in this study, all academic work completed for CS1705 and CS1706 will be collected and evaluated by the investigators during this study. This includes (but not limited to):
  - Graded results from homework and quizzes
  - Graded exams
  - Auto-graded results from take-home projects
  - Hand-graded results from take-home projects
  - Auto-graded results from closed lab assignments
  - Overall course grades

  All coursework collected by this portion of the investigation will be stored securely (either electronically or physically) and viewable only by the course instructor, additional teaching assistants (as needed to perform their traditional activities), the project investigators, or the student who complete the work. Upon completion of the dissertation writing process, all data will be returned to the course instructor for disposal or further storage, as per department/university policy.
- ***Collection and use of anecdotal data***
  Anecdotal data may also be collected during this investigation. This will come in the form of e-mails (both solicited and unsolicited) submitted to the investigators by the students, informal verbal anecdotes related to the study (in and out of the classroom), and transcriptions of group electronic communication mediums employed normally in the administration of the course (LISTSERVs and web-based message boards). All these data will be destroyed upon completion of the dissertation writing process.

III. Risks
There are less than minimal risks to the study participants. The involvement and performance of students in this research is strictly voluntary. A small number of participants may experience eye strain from using a computer screen, or uncomfortable feelings from being watched or interviewed about their experiences in the courses.

IV. Benefits
While there are no direct benefits to you from this research (other than payment for completion of assessment scales and survey), you will find the results of this study will further educate professors the learning styles of students which will in turn allow them to develop appropriate methods for teaching object-oriented design to individuals with vary levels of design aptitude.

You may contact the investigators at a later time for a summary of the research results.

V. Extent of Anonymity and Confidentiality
No one other than investigators will have access to assessment scale results and anecdotal data; both investigators and professors will have access to pre/post survey results. Anyone requesting this data will have access without express written consent from you. Likewise, no printed or electronic rendition of the information that could be directly attributed to you, or any other participant, will be available to anyone other that the investigators without additional written consent from you. Otherwise, and presentation of this research will replace participant names, with anonymous codes or names and/or will report data in summarized form only. Any visual data included in professional presentations and publications will be used anonymously. No information identifying participants will accompany visual material.

VI. Compensation
Participation in this study beyond the pre/post survey (these are required by the professor for all students enrolled in the course) may result in a small monetary stipend or free pizza and soda, consistent for all participants pending funding.

VII. Freedom to Withdraw
You are free to withdraw from a study at any time without penalty. If you choose to withdraw, you will be compensated for the portion of the time of the study (if financial compensation is involved). If you choose to withdraw, you will not be penalized by reduction in points or grade in a course. You are free not to answer any

questions or respond to experimental situations that you choose without penalty.

VIII. Approval of Research
This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Virginia Polytechnic Institute and State University and by the Department of Computer Science.

IX. Subject's Responsibilities
As outlined above, if you agree to participate, your responsibilities will include:

- Initial demographic assessment (performed by the entire class)
- Design readiness assessment scale
- Mid-semester Assessment
- Informal interviews/discussions
- End of year assessment (performed by the entire class)

X. Subject's Permission

I have read and understand the Informed Consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent:

____ __________
Signature Date

Should I have any pertinent questions about this research or its conduct, and research subjects' rights, and whom to contact in the event of a research-related injury to the subject, I may contact:

Investigator: Tracy L. Lewis Ph.D. Candidate Department of Computer Science	Phone: (540)961-1244 Email: tracyL@vt.edu
Investigator: Manuel Pérez-Quiñones Professor, Department of Computer Science	Phone: (540)231-2646 Email: perez@vt.edu
Investigator: MaryBeth Rosson Professor, Penn State – Department of Information Science and Technology	Email:mrosson@psu.edu
Review Board: David M. Moore Research Compliance Office, CVM Phase II (0442)	Phone: (540)231- 4991 Email: moored@vt.edu

[NOTE: Subjects <u>must</u> be given a complete copy (or duplicate original) of the signed Informed Consent.]

Radford University
IRB Proposal
for

Research Study of
A Measure of Design Readiness: The Effects of Individual Differences on Learning Object-Oriented Design

Submitted by
Tracy L. Lewis, Ph.D. Candidate - Virginia Tech
Manuel Pérez-Quiñones, Professor – Virginia Tech
Joseph Chase, Professor – Radford University

Department of Information Technology
Radford University

Department of Computer Science
Virginia Polytechnic Institute and State University

Request for Expedited Review

Title of Study: **A Measure of Design Readiness: The Effects of Individual Differences on Learning Object-Oriented Design**

Names of all investigators (including at least one faculty member): **Tracy L. Lewis, Manuel Pérez-Quiñones, Joseph Chase**

Signatures of all investigators: ____

____

1. Will children, prisoners, residents of institutions, or individuals with cognitive impairments be included as subjects in the proposed study?

Yes No

2. If the proposed research involves the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures, or observation of public behavior, will the information obtained be recorded in such a manner that human subjects can be identified directly or through identifiers linked to the subject?

Yes No N/A

3. For research involving the use of educational tests, survey procedures, interview procedures, or observation of public behavior, could disclosure of the human subjects' responses reasonably place the subject at risk of criminal or civil liability or be damaging to the subjects' financial standing, employability, or reputation?

Yes No N/A

4. Is there more than minimal risk involved in participating in this study?

Yes No N/A

5. What is your rationale for requesting an exemption from full committee review? See "What research needs to be reviewed" in the IRB Policies and Procedures Guide.

   1. Research will be conducted in established or commonly accepted educational settings, involving normal educational practices.
   2. Research will involve the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures or observation of public behavior, **unless** the subjects can be identified directly or through identifiers linked to the subjects **and** disclosure of responses could reasonably place the subjects at risk or criminal or civil liability or be damaging to the subjects' financial standing , employability or reputation.

6. Describe in detail the methodology of your study. (i.e., how will the study be conducted from start to finish, as far as human subjects are concerned? Be specific about the methods, instrumentation, types of data collected, etc.)

   See Attached Document.

7. How will you obtain the informed consent of the subjects? (i.e., how, where, and when will the study be explained to subjects? How will subjects indicate their consent?)

   The informed consent and an explanation of the study will be provided on the first day of classes for the spring 2004 academic semester.

8. What measures will be taken to maintain the confidentiality of information provided by subjects? (i.e., how will the data be stored, who will have access to the data? will the names of subjects be linked to specific items of information?)

   All data will be store on a password protected external 80.0 gigabyte hard drive and only the researchers will have access to these data.

9. Please include a copy of all questions to be contained in questionnaires, surveys, or interviews to be administered.

See Attachments.

Return two copies to:
Janet Hahn
Office of Sponsored Programs & Grants Management
Box 6926 707 E. Main Street
Radford University
Radford, VA 24142
540-831-5035
FAX: 831-6636

Justification of Project

This proposed research will evaluate effective measures of assessing a student's ability to learn object-oriented design (*design readiness*). In particular, we will examine the effectiveness, reliability, and validity of a design readiness assessment scale aimed to measure the design aptitude of undergraduate computer science students.

Measuring one's ability to comprehend and effectively use object-oriented design (OOD) concepts is a complex task in the field of computer science. Although most computer science students can be *taught* to use OOD concepts, very few have mastered these techniques in such a way that would allow them to reuse the techniques in other areas of academic studies. The fundamental problem lies in one's concrete vs. abstract perception abilities. The goal of this study is to identify those areas of individual differences - gender, classification, experience, learning style, design readiness level - that can be utilized by OOD professors in developing robust teaching material and curricula aids.

Data collected by the proposed project will be used by Tracy L. Lewis in support of her dissertation in the department of Computer Science at Virginia Tech.

Procedures

This study will be executed throughout the spring 2004 academic semester, encompassing approximately 120 students enrolled in. The course will be taught by Professor Joseph Chase. Dr. Chase is a primary research investigator on this project and has been consulted in the design and implementation of the requirements of this study as related to course administration.

Participation in this study involves the following:

- Initial demographic assessment (performed by the entire class)
- Design readiness assessment scale
- Group Embedded Figures Test
- Design Development Tasks
- Standardized Cognitive Assessment Tests
- Mid-semester Assessment
- Informal interviews/discussions
- End of year assessment (performed by the entire class)

Selection of Participants

As a part of regular course instruction and assignments, all students enrolled in ITEC 220 – Principles of Computer Science II will be given the opportunity participate this research. All students are encouraged, but required to participate; an alternate assignment will be available for those students opting not to participate. Participation in this research is strictly voluntary and will have no adverse affect on a student's grade.

All participants will complete the Group Embedded Figures Test (GEFT) and several assessment tests created by the Educational Testing Service (ETS) to measure abstract thinking ability, followed by the Design Assessment Scale (created specifically for this research), as well as a Design Development Test.

Group Embedded Figures Test (GEFT)

This is a commercially available test timed test that will take approximately 20 minutes and requires participants to find common geometric shapes in a larger design—this simple assessment yields a wealth of information about field dependence-independence. The Group Embedded Figures Test (GEFT) was developed for research into cognitive functioning, but it has become a recognized tool for exploring analytical ability, social behavior, body concept, preferred defense mechanism and problem solving style as well as other areas. The GEFT is a 25 item assessment contained in a 32 page non-reusable booklet. We are in the process of purchasing the test now.

Educational Testing Service (ETS) Cognitive Tests

One of four tests in a series of tests designed to measure reasoning and other cognitive processes. The test we will administer uses "surface development" tasks that involve imagining the result of folding up a flat pattern into a three-dimensional object. This test is useful in occupations in the engineering, craft, design, or construction industry.

Design Readiness Assessment

Participants will be provided with a multiple choice design readiness scale and asked to answer each question to the best of their ability. The questions are based on seven general principles of object oriented design. The correctness of the student's response will not affect his/her grade in the course.

Design Development Task

Each participant will be asked to complete the design of a "Terrarium" game. The participants will be given a stack of note cards, paper, pens, and pencils, and asked to complete the design of the Terrarium. This process will be video-taped and according to the guidelines described in the attached "Design Task Grading Rubric". This same task will be administered at the beginning and end of the course. The correctness of the student's response will not affect his/her grade in the course.

Collection and use of self-assessment data

Several assessment measures will be used through this investigation.

Initial Demographics Survey

During the first week of courses all students enrolled in ITEC 220 – Principles of Computer Science II will complete an online survey to obtain various demographic data – programming experience, general computer skills/proficiency information, general object-oriented programming and design questions, and computer use confidence. Students will be instructed that answering questions incorrectly will not have an adverse effect on his/her grade. Completion of the survey counts for one homework point, regardless of the correctness of their answers. These data will be used in aggregate form as a part of Tracy L. Lewis' dissertation as well as provide a mechanism for professors and teaching assistants to partner the students in the closed lab activities.

Mid-Semester Cognitive Assessment

Approximately halfway into the course – just before midterms – we will conduct a short survey of those students that volunteered to participate in this experiment. The survey

will assess their perception of their current status in the course and their understanding of various object-oriented programming and design concepts.
Data collected by this portion of the project will be kept confidential by the investigators, protecting the student's anonymity. The assessments (both demographic and mid-year) will be correlated with other student-specific data collected in this investigation, but the resulting data will be anonymous. The assessment data will be destroyed upon completion of this dissertation.

End of year assessment
During the final week of course, all students enrolled in ITEC 220 – Principles of Computer Science II will be asked to complete an end of the year self-assessment survey on their knowledge of object-oriented programming and design. Students will be instructed that this survey will count as one homework grade, regardless of the correctness of their answers.

Collection and use of academic grades
For the participants in this study, all academic work completed for ITEC 220 – Principles of Computer Science II will be collected and evaluated by the investigators during this study. This includes (but not limited to):

- Graded results from homework and quizzes
- Graded exams
- Auto-graded results from take-home projects
- Hand-graded results from take-home projects
- Auto-graded results from closed lab assignments
- Overall course grades

All coursework collected by this portion of the investigation will be stored securely (either electronically or physically) and viewable only by the course instructor, additional teaching assistants (as needed to perform their traditional activities), the project investigators, or the student who complete the work. Upon completion of the dissertation writing process, all data will be destroyed.

Collection and use of anecdotal data
Anecdotal data may also be collected during this investigation. This will come in the form of e-mails (both solicited and unsolicited) submitted to the investigators by the students, informal verbal anecdotes related to the study (in and out of the classroom), and transcriptions of group electronic communication mediums employed normally in the administration of the course (LISTSERVs and web-based message boards). No data will be directly associated with a student. All these data will be destroyed upon completion of the dissertation writing process.

Payment for participation
Participation in this study is encouraged, but not required of students in ITEC 220 – Principles of Computer Science II. For those students opting not to participate, an alternative assignment will be available. Completion of this final survey or the alternative assignment will result in one homework grade. Students will not be penalized for incorrect answers on the survey or alternative assignment.

Participation in solicited focus groups may result in a small monetary stipend or free pizza and soda, consistent for all participants, pending funding.

Risks

There are less than minimal risks to the study participants. The involvement and performance of students in this research is strictly voluntary. A small number of participants may experience eye strain from using a computer screen, or uncomfortable feelings from being watched or interviewed about their experiences in the courses.

Benefits

While there are no direct benefits to the participants from this research (other than payment for completion of assessment scales and survey), the participant will find the results of this study will further prepare professors on appropriate methods for teaching object-oriented design to individuals with varying levels of design aptitude.

Confidentiality and Anonymity

No one other than investigators will have access to assessment scale results and anecdotal data. No printed or electronic rendition of the information that could be directly attributed to a participant will be available. Likewise, any presentation of this research will replace participant names, with anonymous codes and/or will report data in summarized form only. Student information will be stored in a password protected database. The overall data will not have names associated with it; each student's information will be stored using a four digit code – known only by the researchers. Any visual data included in professional presentations and publications will be used anonymously. No information identifying participants will accompany visual material.

Informed Consent

Enclosed.

Data Collections Model

Enclosed.

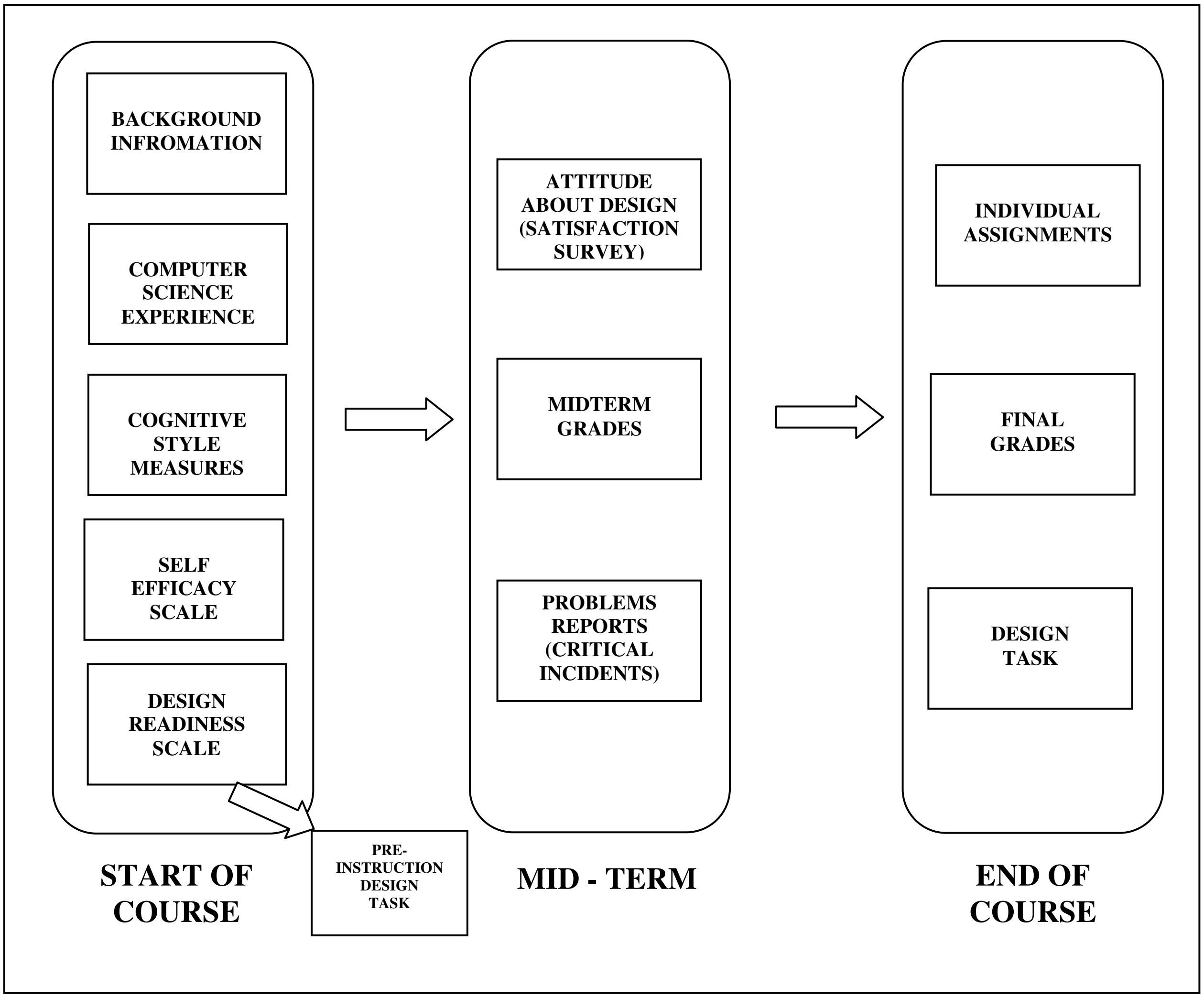


Figure 1 DATA COLLECTION MODEL

Radford University
Informed Consent for Participants in Research Projects Involving Human Subjects

Title of Project: A Measure of Design Readiness: The Effects of Individual Differences on Learning Object-Oriented Design

Investigator(s): Tracy L. Lewis, Ph.D. Candidate – Virginia Tech; Dr. Manuel Pérez-Quiñones, Professor – Virginia Tech; Dr. Joseph Chase, Professor – Radford University

I. Purpose of this Research/Project
You are invited to participate in a research project that evaluates effective measures of assessing a student's ability to learn object-oriented design (*design readiness*). In particular, we will examine the effectiveness, reliability, and validity of a design readiness assessment scale aimed to measure the design aptitude of undergraduate computer science students.

Measuring one's ability to comprehend and effectively use object-oriented design (OOD) concepts is a complex task in the field of computer science. The fundamental problem lies in one's concrete vs. abstract perception abilities. The goal of this study is to identify those areas of individual differences - gender, classification, experience, learning style, design readiness level - that can be utilized by OOD professors in developing robust teaching material and curricula aids.

II. Procedures
This research project will solicit participation of students from ITEC 220 – Principles of Computer Science II. Participation is strictly voluntary and choosing not to participate will not have an adverse affect on your grade. If you agree to participate you could be asked to take part in the following research activities:

- ***Initial Demographics Survey***
  During the first week of courses all students enrolled in ITEC 220 – Principles of Computer Science II will complete an online survey to obtain various demographic data – programming experience, general computer skills/proficiency information, general object-oriented programming and design questions, and computer use confidence. Students will be instructed that answering questions incorrectly will not have an adverse effect on his/her grade. Completion of the survey counts for one homework point, regardless of the correctness of their answers. These data will be used in aggregate form as a part of Tracy L. Lewis' dissertation as well as provide a mechanism for professors and teaching assistants to establish partners in the closed lab activities.
- ***Group Embedded Figures Test (GEFT)***
  This is a commercially available test timed test that will take approximately 20 minutes and requires participants to find common geometric shapes in a larger design—this simple assessment yields a wealth of information about field dependence-independence. The Group Embedded Figures Test (GEFT) was developed for research into cognitive functioning, but it has become a recognized tool for exploring analytical ability, social behavior, body concept, preferred defense mechanism and problem solving style as well as other areas. The GEFT is a 25 item assessment contained in a 32 page non-reusable booklet.
- ***Educational Testing Service (ETS) Cognitive Tests***
  One of four tests in a series of tests designed to measure reasoning and other cognitive processes. The test we will administer uses "surface development" tasks that involve imagining the result of folding up a flat pattern into a three-dimensional object. This test is useful in occupations in the engineering, craft, design, or construction industry.
- ***Design Readiness Assessment Scale***
  Participants will be provided with a multiple choice design readiness scale and asked to answer each question to the best of their ability.
- ***Design Development Task***
  Each participant will be asked to complete the design of a "Terrarium" game. The participants will be given a stack of note cards, paper, pens, and pencils, and asked to complete the design of the Terrarium. This process will be video-taped and according to the guidelines described in the attached "Design Task Grading Rubric". This same task will be administered at the beginning and end of the course. The correctness of the student's response will not affect his/her grade in the course.

- ***Mid-Semester Cognitive Assessment***
  Approximately half into the course – just before midterms – we will conduct a short survey of those students that volunteered to participate in this experiment. The survey will ask assess their perception of their current status in the course and their understanding of various object-oriented programming and design concepts.
  Data collected by this portion of the project will be kept confidential by the investigators and will only be shared with the course instructor in an anonymous fashion, protecting the student's anonymity. The assessments (both demographic and mid-year) will be correlated with other student-specific data collected in this investigation, but the resulting data will be anonymous. The assessment data will be destroyed upon completion of this dissertation.
- ***End of year assessment***
  During the final week of courses, all students enrolled in ITEC 220 – Principles of Computer Science II will be asked to complete an end of the year self-assessment survey on their knowledge of object-oriented programming and design. Students will be instructed that this survey will count as one homework point, regardless of correctness of answers to questions requiring knowledge of object-oriented programming and design.
- ***Collection and use of academic grades***
  For the participants in this study, all academic work completed for ITEC 220 – Principles of Computer Science II will be collected and evaluated by the investigators during this study. This includes (but not limited to):
    - Graded results from homework and quizzes
    - Graded exams
    - Auto-graded results from take-home projects
    - Hand-graded results from take-home projects
    - Auto-graded results from closed lab assignments
    - Overall course grades

  All coursework collected by this portion of the investigation will be stored securely (either electronically or physically) and viewable only by the course instructor, additional teaching assistants (as needed to perform their traditional activities), the project investigators, or the student who complete the work. Upon completion of the dissertation writing process, all data will be destroyed.
- ***Collection and use of anecdotal data***
  Anecdotal data may also be collected during this investigation. This will come in the form of e-mails (both solicited and unsolicited) submitted to the investigators by the students, informal verbal anecdotes related to the study (in and out of the classroom), and transcriptions of group electronic communication mediums employed normally in the administration of the course (LISTSERVs and web-based message boards). No anecdotal data collected will be directly attributed to you. All these data will be destroyed upon completion of the dissertation writing process.

III. Risks
There are less than minimal risks to the study participants. The involvement and performance of students in this research is strictly voluntary. A small number of participants may experience eye strain from using a computer screen, or uncomfortable feelings from being watched or interviewed about their experiences in the courses.

IV. Benefits
While there are no direct benefits to you from this research (other than payment for completion of assessment scales and survey), you will find the results of this study will further educate professors the learning styles of students which will in turn allow them to develop appropriate methods for teaching object-oriented design to individuals with vary levels of design aptitude.

You may contact the investigators at a later time for a summary of the research results.

V. Extent of Anonymity and Confidentiality
No one other than investigators will have access to assessment scale results and anecdotal data. No printed or electronic rendition of the information that could be directly attributed to you, or any other participant, will be available to anyone. Likewise, any presentation of this research will replace participant names, with anonymous codes or names and/or will report data in summarized form only. Any visual data included in professional presentations and publications will be used anonymously. No information identifying participants will accompany visual material.

VI. Compensation
Participation in this study is encouraged, but not required of students in ITEC 220 – Principles of Computer Science II. For those students opting not to participate, an alternative assignment will be available. Completion of this final survey or the alternative assignment will result in one homework grade. Students will not be penalized for incorrect answers on the survey or alternative assignment.

Participation in solicited focus groups may result in a small monetary stipend or free pizza and soda, consistent for all participants, pending funding.

VII. Freedom to Withdraw
You are free to withdraw from a study at any time without penalty. If you choose to withdraw, you will be compensated for the portion of the time of the study (if financial compensation is involved). If you choose to withdraw, you will not be penalized by reduction in points or grade in a course. You are free not to answer any questions or respond to experimental situations that you choose without penalty.

VIII. Approval of Research
This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Radford University and by the Department of Information Technology.

IX. Subject's Responsibilities
As outlined above, if you agree to participate, your responsibilities will include:

- Initial demographic assessment (performed by the entire class)
- Design readiness assessment scale
- Mid-semester Assessment
- Informal interviews/discussions
- End of year assessment (performed by the entire class)

X. Subject's Permission

I have read and understand the Informed Consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent:

____ __________
Signature Date

Should I have any pertinent questions about this research or its conduct, and research subjects' rights, and whom to contact in the event of a research-related injury to the subject, I may contact:

Investigator: Tracy L. Lewis Ph.D. Candidate, Department of Computer Science Virginia Tech	Phone: (540)818-8010 Email: tracyL@vt.edu
Investigator: Manuel Pérez-Quiñones Professor, Department of Computer Science Virginia Tech	Phone: (540)231-2646 Email: perez@vt.edu
Investigator: Joseph Chase Department Head, Professor, Department of Information Technology	Phone: (540)831-5997 Email: jchase@radford.edu

APPENDIX B

COURSE SYLLABI

Intro to Object Oriented Design II, Spring 2004

Course Description

Detailed coverage of data structures, algorithms, and the methods of object-oriented design and software construction. Basic concepts in human-computer interfaces and graphics. Design and construction of medium-sized object-oriented programming projects with an emphasis on teamwork and software engineering.
Learning Objectives: Having successfully completed this course, the student will be able to:
Design, implement, test, and debug programs using dynamic data structures such as linked lists, stacks, and queues;
Design, implement, and test medium-sized programs (e.g., 1K-3K lines of code), including network-based and interactive applications.
Design, implement, and test reusable components as part of a medium-sized object-oriented program;
Use a range of software tools (e.g. editors, class browsers, bug tracking, debuggers) in development of a medium-sized software product.

Prerequisites

C or better in CS1, no exceptions.

First Day Attendance

First day attendance is required. If you miss the first day, you will be dropped from the course. If you cannot attend class the first day, contact the professor **before class starts**.

Web page

This page will be updated throughout the semester with information on deadlines, changes to the schedule, presentations, etc. The web page includes several dynamic features that should help you get the most out of this course. Some of these include: discussion board, grade lookup, semester calendar, and a very simple search engine over transparencies.

Attendance and Participation

Attendance at class is necessary for successful completion of the course. Attendance is particularly important on specials days, such as homework-due days, exam days, etc. However absences will not count against your grade, but are discouraged unless special circumstances exist.
Attendance to your assigned lab section is required. You will have a deliverable due at the end of each lab section, so if you skip a lab you will get a zero in that week's lab. You are required to attend the lab that you are registered for. Attendance to another lab without prior permission counts as an abscence.
Homeworks/Projects due in class/lab are due at the beginning of the meeting period and will not be accepted late. This includes handing in your work on your way out of class, or turning them when you arrive

> **NOTE** that it is your responsibility to turn in the required work at the assigned due date, it is NOT the responsibility of the professor or the GTAs to pick it up or to remind you to turn it in.

There might be some participatory exercises done in class. If you are asked to participate in these, it is expected that you will do so.
Students are expected to read the assigned material prior to class, check the web page for the assigned readings and their dates. Some class time will be used for lectures, but attending lectures will not be sufficient for full understanding of the concepts from the readings.

Assigned Work

Throughout the semester you will have several assignments of different kinds. Each might require different skills from you, and each will require different amount of effort. In general you can count on the following:
Quizzes - There might be some quizzes in the course. These might be in class quiz (announced and unannounced) as well as online quizzes.
Homeworks and Quizzes - There might be some quizzes in the course. These might be in class quiz (announced and unannounced) as well as online quizzes. The Homeworks are shorter individual assignments that are due in a short period of time (one to three classes later). Some of these are assigned ahead of time (check the calendar on the web), others might be assigned with just enough time to complete them.
Programming Projects - there will be several programming projects. You are expected to work on these individually and submit it using the Web-Cat system (much like you did in CS 1705).
Others - there might be other exercises that will earn you credits towards your final grade. These vary from semester to semester, so attend class everyday so you find out about these "freebies". One such freebie is participation in experiments going on in the department. These often are replacement of other grades, or for dropping a lower grade.

Quizzes

In-class quizzes and online quiz will be used throughout the semester. There will be no make up of quizzes, not even with a medical excuse.
Quizzes are not going to be discussed in class and you might not get the quiz back other than finding your score and the key on the web.

Homeworks

There will be a number of homework assignments throughout the semester. It is possible, but not guaranteed, that the lowest homework score will be dropped. Most homework assignments will be submitted electronically. No late homeworks will be accepted. There will be no make up of homework assignments, not even with a medical excuse.

Programming Projects

There will be several programming projects in the semester. For some of these, you will have a chance to "revise" the project once you get your initial grade, with the possibility of improving your grade. This will be done by taking advantage of peer reviews. More info will be given in class.
Each project part has a firm due date. This is the date by which you have to submit your project or risk losing points. Late programming projects can be submitted, but have a 10% penalty per day late. No projects will be accepted more than three days late. The 10% is deducted from your the score you obtain. For example, if you turn in a project

late and your score is 80 points. 10% for late penalty will mean that you will lose 8 points, and your score will be 72.

> **IMPORTANT** - You must turn in all projects in order to pass the course. The projects might build on previous ones, so missing one will might make others more difficult. Do not miss projects.

Before you start begging for extensions to project schedules, consider the following:
The assigned time to do the projects is given considering the time it would take you to do the project and the timing of when the relevant material is discussed in class.
If we "delay" a project deliverable, that means that you will probably have that you will probably have less time to do the other projects. It is in your interest to get them done in time.

Exams

The tests are to be done at class time and include three main sections. The first is to test your declarative knowledge of terminology, concepts, etc. The second section of the test has several short problems. The last section has a larger design problem, often related to a programming project you have already done.
No makeups of exams are given. However, if you have to miss an exam, you have to contact the professor before the exam and provide a medical excuse for missing it. **Note** that I do not consider a visit to the Health Center a valid excuse.
If you miss the first exam and **have a valid excuse** (that is, checked with me ahead of time), exam 2 will count as double (exam 2 is harder, so don't miss the first).
If you take the first but miss the second exam and **a valid excuse**, the final exam will count for the 2nd and final exam.
If you miss both exams (1 and 2, valid excuse or not) you will get an F in the course.

Absence, Makeups, and other special circumstances

No makeups are allowed in this course. Homeworks not turned in on time will not be accepted. Quizzes missed get a zero. Projects can be turned in late up to three days with the appropriate penalty. Exams missed can be "replaced" under special circumstances as explained above.
Consider the following observations:
I will give extensions only for the amount of time that you lost due to sickness. So, if you have a bad case of the flu and you were down for a day, you will get an extra day to complete your work. Note that you still need to give me proper medical evidence that you were not able to do your work for that day.
A slip from the Health Center only shows that you went to see the doctor (excusing you of maybe 1 hour time).
Do not leave the projects for the last minute, you are putting yourself at risk of last minute bad luck (Murphy's law).
If you run into the unfortunate situation that requires an extension, know how much time you need. Don't come to me saying "I need an extension" because I have not seen your work, so I do not know how much longer to give you. And I will not extend deadlines for weeks at a time. Most likely I will give you one more day, and of course *only if you have the proper evidence*.
In case of the unfortunate situation that you have a trip out of town to go see the President of the United States because you are being honored at the White House (or some other activity of that magnitude), I can give you a makeup exam, but it must be

ahead of time. Note, however, that these special time exams are different than the exams that the rest of the class will take. So, just call the President and tell him to give you the award in the Summer.
Oh, by the way, your machine crashing, getting attacked by a virus, updating your OS and in the process losing some data, and other technology-based excuses are not considered valid. I consider these as the new millennium version of "the dog ate my homework", so plan for these unfortunate situations, they will happen all the time. Make backups frequently, and keep a copy of your backup at a separate location (like you significant-other's dorm, or filebox.vt.edu).

Grading

Your grade will be based on the scores you obtain on your work. There will be no curve applied so your scores, so be sure to study and work hard for every single assignment and test. Your work will be weighted as follows:

Quizzes and homeworks	10%
Laboratories	10%
Programming projects (several, equal weight)	35%
Two exams and a final exam (each 15%)	45%
Total	100%

Final grades will be set according to the usual 10-point scale using A, B, C, D and F. I reserve the right use the extended scale (A, A-, B+, etc.). I do not plan to use a curve, so do not count on getting 88 and waiting for the curve to pull you through. It won't. Study to get a 100. All the scores are rounded to one decimal place and the final score rounded to integers (i.e. 89.4 is a B and 89.5 is an A).

Honor Code

With the exception of lab work that might be done in pairs, all other work on assignments and exams is to be your own. You will be required to sign an honor code statement on all individual work.
What is plagiarism? Check the website, http://www.plagiarism.org/. I do not tolerate plagiarism, so avoid doing it and do not even try to justify it by giving excuses that begin as "I was not aware that ..."

Special Needs

If you have any special needs or circumstances (disability accommodations, religious holidays, etc.) please see the instructor during office hours. Please do so early in the semester, so we can plan for accommodations for exams and quizzes with plenty of time.

Calendar for CS 1706: Intro to Object Oriented Design II

Date	Material/Related Links	Readings
Mon 1/19	Intro to course, OO Review, classes, inheritance	LC Appendix, H1, LC1
Wed 1/21	Experiment in class	
Mon 1/26	University closed due to snow	
Wed 1/28	Evaluation of class designs - tradeoffs and reuse, encapsulation	H3
Mon 2/2	Collections, array implementations	LC2
Wed 2/4	Linked structures, encapsulation	LC3
Mon 2/9	Lists, linked lists, other nonlinear structures	
Wed 2/11	OO Design process	H2
Mon 2/16	UML, CRC, Relationships in oo design	
Wed 2/18	More on OO design	
Mon 2/23	Exam 1	H1-3, LC1-3, LC Appendix
Wed 2/25	Recursive programming	LC4
Mon 3/1	More on recursion, Stacks	LC6
Wed 3/3	Stacks	LC6, in class Quiz
Mon 3/8	Spring break	
Wed 3/10	Spring break	
Mon 3/15	More on Stacks, Queues	LC7
Wed 3/17	Queues, search, lists	LC7, LC8
Mon 3/22	Search, Interfaces	LC 5.1, H4
Wed 3/24	Interface types, polymorphism	H4
Mon 3/29	Lists, Graphical User Interfaces	LC8, H5
Wed 3/31	More on GUIs	H5, some of H6
Mon 4/5	Exam 2 Part I	Part 1 of Exam 2, multiple choice, short answers. Take home given for wednesday.
Wed 4/7	Exam 2 Part 2	OO Design and Coding, in class, based on take home, **Open Book!**
Mon 4/12	Inheritance and interfaces	H6
Mon 4/19	More on Inheritance	H6
Wed 4/21	Painting and Drawing in Java	
Mon 4/26	Java Object Model, Reflection	H7
Wed 4/28	Frameworks	H8
Mon 5/3	Lists	LC8
Wed 5/5	Sorting	LC 5.2
Mon 5/10	Final exam	

Principles of Computer Science II

Prerequisite: CS 1 (with a grade of C or better)
Students who have received credit for CPSC 124 may not receive credit for the current course.

Notes, Assignments, and Labs:
The course notes are provided so that you may concentrate on listening to and participating in the discussions in class. The Assignments and Lab Assignments for this course will be posted as we progress through the course. All of the code presented in the text is also available here as a zip file for download.

Course Notes:	Spring 04 Assignments:	Spring 04 Lab Assignments:
Chapter 1	Assignment 1	lab1
Chapter 2	Assignment 2	lab2
Chapter 3	Assignment 3	lab3
Chapter 4	Assignment 4	lab4
Chapter 5		lab5
Chapter 6		lab6
Chapter 7	Extra Credit	lab7-lab amnesty week
Chapter 8		lab8
Chapter 9		lab9
Chapter 10		lab10
Chapter 11		lab11-12
Chapter 12		
Chapter 13		
Chapter 14		
Appendix A		
Ethics		code presented in the text

Old Notes, Assignments, and Labs:
Notes, assignments, labs and other materials from previous semesters are provided as additional material for your benefit.
Course Description:
Continuation of the development of a disciplined approach to programming, with emphasis on data abstraction. This course emphasizes the design and implementation of solutions to problems which require complex data structures. The topics include:

Programming Fundamentals
- multi-dimensional arrays
- recursion

Java Topics
- interfaces and abstract classes
- applets

- applications
- inner classes

Graphical User Interface
- events
- listeners
- components

Data Structures
- stacks
- queues
- vectors
- lists
- binary tree concepts
- binary search tree concepts

Recursive Sorting and Searching Concepts
- quicksort
- mergesort
- binary search (arrays and trees)

OO Topics
- objects
- references
- classes, methods, fields
- instance v.s. class members
- inheritance
- polymorphism
- peristence
** serialization*
** marker interfaces*
- overriding

Software Engineering
- Problem solving
- Software Analysis and Design
- Testing and debugging
- Documentation and program structure
- UML
- encapsulation
- Abstraction and Data Structures

Language Topics
- linked structures
- recursion
- exceptions

Recurring Theme
- Analysis of Algorithms

Program examples for some problems will be introduced by the instructor, students will then be required to finish some projects for problems similar to those discussed in classes. Students will progressively learn more advanced techniques such as algorithm design and development, data structures, and the application of software engineering methods from the lectures and programming exercises. The focus of this course is problem solving NOT programming! However, students will implement solutions to problems in order to prove that their solutions work.

Text:

The text for this course is Java Software Structures by Lewis and Chase. A website (**http://www.radford.edu/~jchase/jss.html**) is available for you to provide feedback should you find errors or wish to make suggestions about the text. Your input is most welcome.

Grading:

Midterm Exam	25%
Final Exam	35%
Quizzes and in-class assignments	5%
Projects	25%
Lab Projects	10%

· All assignments will be graded on correctness and quality. Satisfying the minimum requirements will result in a grade of "C"!
· Late assignments will NOT be graded and will result in a grade of 0!
·
Attendance Policy:
Attendance will be taken in every lecture and every lab. Every third absence will result in a 10 point penalty on your final grade. For example, students missing six classes will have 20 points subtracted from their final grade. Attendance sheets will be available in both lecture and lab sessions and it is the students responsibility to make sure their attendance is recorded. Signing another persons name on the attendance sheet is an Honor Code violation!
Honor Code:
The will be strictly enforced!

APPENDIX C

DEMOGRAPHICS SURVEY

DESIGN READINESS STUDY
BACKGROUND INFORMATION

INSTRUCTIONS:
Please complete the following background information to the best of your ability.

LAST FOUR DIGITS OF YOUR SSN: _______________

1. Gender:
☐ Male ☐ Female

2. Age:
☐ under17 ☐ 17
☐ 18 ☐ 19
☐ 20 ☐ 21-24
☐ 25 -29 ☐ 30 or older

3. Classification:
☐ Freshman ☐ Sophomore
☐ Junior ☐ Senior
☐ Graduate Student

4. Major:
☐ Computer Science
☐ Computer Engineering
☐ Industrial Systems Engineering
☐ Business
☐ Other:

5. English as native language:
☐ Yes ☐ No

6. Enrollment Status:
☐ Full-time ☐ Part-time

7. High school grade point average:
☐ below 2.0 ☐ 2.0 – 2.49 ☐ 2.5 – 2.74
☐ 2.75 – 2.99 ☐ 3.0 – 3.24 ☐ 3.49 – 3.25
☐ 3.5 - 3.79 ☐ 3.8 – 4.0 ☐ above 4.0

8. College grade point average:
☐ below 2.0 ☐ 2.0 – 2.49 ☐ 2.5 – 2.74
☐ 2.75 – 2.99 ☐ 3.0 – 3.24 ☐ 3.49 – 3.25
☐ 3.5 - 3.79 ☐ 3.8 – 4.0 ☐ above 4.0

9. Number of Computer Science and/or programming courses taken in high school:
☐ none ☐ 1 ☐ 2
☐ 3 ☐ 4 ☐ 5

10. Number of Advanced Placement CS courses taken:
☐ none ☐ 1 ☐ 2
☐ 3 ☐ 4 ☐ 5

11. Took Computer Science Advanced Placement Exam:
☐ No
☐ Yes

If YES, Please list Score: ___________

COMPUTER EXPERIENCE

INSTRUCTIONS: Please respond to the following statements by checking the box that most accurately describes your computer experience.

1. Knowledge of computers:

- ☐ very minimal
- ☐ limited
- ☐ average
- ☐ very knowledgeable
- ☐ extensive knowledge

2. Work experience in computer science related field:

- ☐ less than 1 yr
- ☐ 1 – 2 years
- ☐ 3-4 years
- ☐ 4 – 5 years
- ☐ more than 5 years

3. I use a computer **for school assignments in any course**:

- ☐ Once a month or less often
- ☐ Several times a week
- ☐ Once a week
- ☐ Once a day
- ☐ Several times everyday

4. I use a computer **for leisure** (i.e. email, chat, etc):

- ☐ Once a month or less often
- ☐ Several times a week
- ☐ Once a week
- ☐ Once a day
- ☐ Several times everyday

5. Please indicate the type of computer package you use **most often**:

- ☐ Wordprocessing packages
- ☐ Spreadsheet packages
- ☐ Database packages
- ☐ Presentation packages
- ☐ Statistics Packages
- ☐ Desktop Publishing
- ☐ Gaming packages
- ☐ Other Multimedia packages
- ☐ Compiler
- ☐ Other: ___________________________________

INSTRUCTIONS: Please respond to the following statements by checking the box that most accurately describes your level of programming knowledge with each of the following languages.

6. Rate yourself on your knowledge of each of the following languages/software applications, using a scale from 1 to 5:

	1 None	2 Novice	3 Intermediate	4 Proficient	5 Expert
C	☐	☐	☐	☐	☐
C++	☐	☐	☐	☐	☐
JAVA	☐	☐	☐	☐	☐
JAVA SCRIPT	☐	☐	☐	☐	☐
JSP	☐	☐	☐	☐	☐
ADA	☐	☐	☐	☐	☐
TCL-TK	☐	☐	☐	☐	☐
HTML	☐	☐	☐	☐	☐
PERL	☐	☐	☐	☐	☐
CGI SCRIPT	☐	☐	☐	☐	☐
COBOL	☐	☐	☐	☐	☐
ASP	☐	☐	☐	☐	☐
VISUAL BASIC	☐	☐	☐	☐	☐
DREAM WEAVER	☐	☐	☐	☐	☐
MACROMEDIA FLASH	☐	☐	☐	☐	☐
MICROSOFT FRONTPAGE	☐	☐	☐	☐	☐
.NET APPLICATIONS	☐	☐	☐	☐	☐
OTHER: ________________	☐	☐	☐	☐	☐
OTHER: ________________	☐	☐	☐	☐	☐

INSTRUCTIONS: Please respond to the following statements by checking the box that most accurately describes your level of experience with each of the following operating systems.

7. Rate yourself on your knowledge of the following computer operating systems using a scale from 1 to 5:

	1 None	2 Novice	3 Intermediate	4 Proficient	5 Expert
WINDOWS	☐	☐	☐	☐	☐
DOS	☐	☐	☐	☐	☐
UNIX	☐	☐	☐	☐	☐
MACINTOSH or APPLE OS	☐	☐	☐	☐	☐
LINUX	☐	☐	☐	☐	☐
OTHER: ________________	☐	☐	☐	☐	☐

INSTRUCTIONS: Please respond to the following statements by checking the box that most accurately describes your level of experience with each of the following statements.

8. Rate yourself on your expertise in the following areas:

	1 None	2 Novice	3 Intermediate	4 Proficient	5 Expert
Experience working with a team	☐	☐	☐	☐	☐
Experience working on large projects (over 2,000 lines of code)	☐	☐	☐	☐	☐
Experience in real-time systems and programming	☐	☐	☐	☐	☐
Experience in database programming and/or database manipulation (Access,Oracle,SQL,etc)	☐	☐	☐	☐	☐
Experience with laptop computers (notebooks)	☐	☐	☐	☐	☐
Experience in programming mobile computing devices (ex. PDA's)	☐	☐	☐	☐	☐
Experience using a wireless network	☐	☐	☐	☐	☐
Experience with network protocols and network architectures	☐	☐	☐	☐	☐
Experience in web design and web programming tools	☐	☐	☐	☐	☐
Knowledge of Object-Oriented Programming (OOP)	☐	☐	☐	☐	☐
Knowledge of Object-Oriented Design (OOD)	☐	☐	☐	☐	☐
Knowledge of Unified Modeling Language (UML)	☐	☐	☐	☐	☐

COMPUTER UNDERSTANDING AND EXPERIENCE (CUE) SCALE

INSTRUCTIONS: Please respond to the following statements by circling the number of the expression that most accurately depicts your opinion.

1. I frequently read computer magazines or other sources of information that describe new computer technology.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

2. I know how to recover deleted or "lost data" on a computer or PC.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

3. I know what a LAN is.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

4. I know what an operating system is.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

5. I know how to write computer programs.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

6. I know how to install software on a personal computer.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

7. I know what e-mail is.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

8. I know what a database is.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

9. I am computer literate.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

10. I regularly use a PC for word processing.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

11. I often use a mainframe computer system.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

12. I am good at using computers.

1	2	3	4	5
Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

APPENDIX D

DESIGN READINESS ASSESSMENT SCALE

Design Readiness Scale

Directions:
The Design Readiness Scale is an assessment of your ability to learn object-oriented design concepts and strategies. You will be given 28 problem scenarios covering various concepts involved in completing an object-oriented design task. These scenarios are based on common life practices, but do not read personal life experiences into any of the scenarios.

The problem scenarios below are followed by questions based on their content. After reading each scenario, reflect on the interaction between the people, events, and objects. Choose the best answer to each question and provide an explanation for your decision. Then choose a second choice solution and provide an explanation for that decision as well. **PLEASE** answer all questions following the scenarios ***only*** on the basis what is ***stated/inferred*** in the scenarios, not on "real-life" experiences.

Example:
Jerry eats liver for dinner every night at 7:00 p.m. and then he watches two hours of television, before reading for one hour, and then he heads to bed for the night. Jerry has work on a big proposal for work tomorrow.

1. When should Jerry prepare the proposal?
    a. Between dinner and his television time
    b. Before he sits down for dinner
    c. Before he heads off to bed
    d. Tomorrow morning

- Best solution: B

The Smith family - Father Bill, Mother Sally, Daughter Sarah and Son Johnny live in suburbia. Father Bill has instructed Son Johnny to take out the trash for pick-up every Monday by 6:00 a.m. and Daughter Sarah should go out and walk the dog every Wednesday at 4:00 p.m. Son Johnny decided not to take the trash out because he would rather walk the dog, so he told Mother Sally of his new plans.

1. **Did Johnny inform the correct person of his decision to switch jobs?**
   a. Yes, as long as he informed one of his parents of his decision.
   b. Yes, his mother will convey Son Johnny's decision to Daughter Sarah.
   c. No, he should have personally informed Daughter Sarah.
   d. No, he should have informed Father Bill of his decision.

The Johnson family is responsible for this week's car pool. Mother Johnson drives the children to school and Father Johnson picks them up after school. Little Timmy Turner rides with the Johnson family, but is sick and won't be attending school. Tim's mother called the Johnson house and spoke with Father Johnson about Timmy's illness. Father Johnson was watching the basketball game and forgot to tell Mother Johnson when she came in the room. The next morning Mother Johnson waited outside of little Timmy's house for 20 minutes, causing her to be extremely late picking up the other children, making them all late for school.

2. **How could the Johnson family have avoided the children being late for school**?
   a. Mother Johnson should have asked dad who was on the phone immediately after he got off the phone.
   b. Father Johnson should have told mother Johnson the news immediately after he got off the phone with little Timmy Turner's mom.
   c. Little Timmy Turner's mother should have asked to speak to mother Johnson because mother Johnson is responsible for picking up the children before school.
   d. Mother Johnson should have buzzed little Timmy's door bell after waiting for five minutes.

Barbara had the biggest 30th Birthday bash Blacksburg had ever seen. So many people showed up with gifts in hand. Barbara needs to send thank-you e-mails to over 200 people. Some of the people are co-workers, some are friends, and others are family members. Several of them bought her similar gifts, where as others bought really unique items. She really wants to thank each person for the specific gift they bought her.

3. **After Barbara has written a list of all her gifts, how should Barbara go about sending out the e-mails to all the people?**
   a. Send them out randomly.
   b. Send them out based on her relationship with the person.
   c. Send them to co-workers and then family.
   d. Send them based on the gifts.

4. In what manner should Barbara send the e-mails to all the people?

a. Send individually e-mails to each person.
b. Send one general e-mail.
c. Group the people and send out group e-mails.
d. Develop a general template e-mail and tailor it for various people.

Several community service organizations would like to start after school programs to tutor math, science, and computer skills. The service organizations are located within the same county, but on four distinct sides of the county. All the schools in the county are in the same school system, accordingly all the students are taught the same information on the same schedule.

5. How should the community service organizations go about organizing the after school program?

a. The four organizations should divide into pairs to handle different zones of the school system.
b. Each organization should meet with the principles of the schools and work with each school to form the after school program.
c. Each organization should run their program separately and handle the portion of the county relative to their location and school proximity.
d. The organizations should set up one community service program that services that entire school system from different locations and assign one person from each organization to serve as a communication liaison between the different sites.

6. Billy will be moving from his current home to the opposite side of the county. He wants to continue receiving help from the after school program. What does he have to do to continue receiving help?

a. Simply show up at the new after school program.
b. Inform his current after school program and they will inform the new program.
c. Continue working with his same after school program.
d. Inform the principal of his school and s/he will inform the new after school program.

If Kelly wants to get a dog, her mom told her she had to do all her chores for one month and get straight A's on her report card. It is one week before report cards are released; Kelly has straight A's, and has been very diligent about her chores. Kelly's mom is really impressed with her dedication. In Kelly's deep preparation for her final exams, she forgot to empty the recycle bins and now they won't get emptied for two more weeks. Kelly brought home all A's on her report card and her mom is ecstatic about her grades.

7. Should Kelly's mom buy her the dog?

a. Yes, Kelly got straight A's on her report card and only missed one chore.
b. Yes, Kelly has proven to be diligent and responsible in school and at home.
c. No, Kelly missed emptying the recycle bins.
d. No, Kelly's mom never really had any intention of buying a dog.

Calvin Rodgers has drawn out the plans for a new 60 story – 900 ft. sky scraper to be built in the heart of Metropolis. There were 60 investors and each investor wanted a private floor to conduct business. Just as Calvin was about to present his work to the investors, the City Clerk's office called and informed him that his plans were 150 ft above the new sky scraper zoning laws.

8. How should Calvin change the plans to meet the new zoning regulations?

a. Simply subtract the 150 ft. from the overall building height of 900 ft
b. Keep the current design and distribute the loss of 150 ft. throughout the floors
c. Reduce the building height by eliminating several floors of the building
d. Start over and redesign the entire building

Randy Thomas is the CEO of the Clarke Corporation. In order to cut costs and increase productivity, Randy has to restructure his organization. Currently the marketing department duties entail designing campaign strategies, the sales department duties entail talking to customers and conveying product likes/dislikes to the product designers, the product design department duties entail the design prototypes, and the product development department duties entail creating the final product.

9. How should CEO Thomas restructure the Clarke Corporation?

a. Take the best representatives from each department and create a new organizational structure
b. Combine the Marketing & Product Design departments
c. Eliminate the Product Design department
d. Combine both Sales & Marketing departments and the Product Design and Product Development departments

Jenny has a million things to do before she catches her plane tomorrow morning. She is remembering that she has to wash her hair, iron her clothes, take the dog to the kennel, write checks for her bills, check the weather, check the status of her flight, call her mom, paint her nails, pack, pick up the rest of her clothes from cleaners, and take out the trash.

10. How should Jenny go about doing all of the things she has to do?

a. Do things as she remembers them.
b. Start with the task that requires the least amount of energy or time and work towards the most time consuming task.
c. Just do things and cross them off the list as she goes through her day.
d. Group them and then take care of them.

Abram wants to build a new television. The new 35 inch high definition television will have picture-in-picture capabilities, child lockout features, and movie reminder options. It will have a sleek silver metal casing to match all the new-age modern furniture. The set has all the necessary components on the inside, now Abram can't figure out where he should put the on/off switch.

11. **How should Abram decide where to put the on/off switch on the new television?**
    a. Think about the location of the on/off switch on his old television and put it there.
    b. Put it in a place that's easily accessible by the television viewer.
    c. Leave the on/off switch off of the television and place it only on the remote control.
    d. Put it in a hidden place on the side of the television.

Billy is a new software development engineer with MacroHard Software Company. His boss wants him to become familiar with designing commercial, software so he assigns Billy the project of designing a calculator.

12. **From the choices provided, what functionality (calculator features/properties) should he consider?**
    a. Screen, input, output, processor
    b. Layout, numbers, addition, subtraction, division
    c. Operations, operands, layout, state information, error recovery
    d. Numbers, operations, execution of equal sign, memory

Diann is the leader of a local book club. It is her turn to host the book club with Candi and Crystal. Diann has been in the book club for almost four years and knows everyone and everything about the operation of the book club. She interviewed Candi and Crystal for membership six months ago. There is less than three weeks before the meeting and they still have to decide on the book for the group to read, find a location for the meeting, order food to serve at the meeting, notify book club members of location/date, and create the discussion agenda.

13. **How should Diann, Crystal, and Candi go about performing all the tasks listed above?**
    a. Allow each person to pick tasks to complete
    b. Diann should randomly assign tasks
    c. Diann should perform all the tasks herself and inform Candi & Crystal about it
    d. Diann should assign tasks based on what she knows about Candi and Crystal.

Maggie is developing an online calendar system for her office. There are seven programmers, two managers and one office clerk. Each person should have write permission on their own calendar; read permission of everyone's calendar; and the office clerk should have write permission on both managers computer.

14. How should Maggie design the permission feature?

a. Design the permission feature such that there should be three operations, each represent each employee type
b. Design the permission feature such that there is only one operation and it performs tasks based on the amount and type of information it receives
c. Design the permission feature such that there are general operations so that you can extend them in the future
d. Design the permission feature such that there is one operation and conditional statements to handle each type of employee

Jessica works at ABC ChildCare Center. She works with the age group that is scheduled to learn shapes and figures next week. Jessica created a drawing tool, that will help children learn shapes, by drawing colored rectangles and circles. Her boss came in and asked her to have her drawing tool draw a red square.

15. Should Jessica's tool be able to draw the red square?

a. Yes, because they are both four sided objects
b. Yes, because a square is a specialized form of a rectangle
c. No, only two specific shapes are specified in the tool.
d. No, Sally didn't identify specific colors in her tool.

Sandra is the store manager of the CornerBlaster video store. Her employees have been complaining about the antiquated video check-out system the store currently uses. As a surprise for her employees she wants to purchase a new system.

16. What features should Sandra look for in the new video check-out system?

a. Inventory database, rental agreement, customer accounts
b. Video database, information on new releases, information on old releases, payment information
c. DVD inventory, VHS inventory, customer information
d. Video database, payment information, customer accounts

David will be graduating from college in a few months and is in the process of looking for a job. Each job requires a cover letter, which should state why his qualifications match the job description, as well as a detailed resume. David has found almost thirty jobs for which he meets qualifications.

17. How should David design the cover letter?

a. Design the cover letter to include all his qualifications and the specific qualification that the employer needs will be covered in the letter
b. Design a generic cover letter and include personal statements to add a personal touch to it
c. Design a general template letter and customize it for every employer
d. Design different cover letters based on his specific qualifications and based on the job requirements, then choose a letter from his set of letters.

Tonya sees a TeleSERVE USA advertisement for an unbelievable deal on a new cell phone. But there isn't a TeleSERVE USA cell phone store in her area. She remembers StereoHut now offers cell phone packages for all the major cell phone providers. Tonya decided to go into StereoHut and talk with a salesperson about getting a TeleSERVE USA phone.

18. How should the salesperson handle Tonya's request?

a. He should try and sell her the best looking phone in the store
b. He should connect her with the TeleSERVE USA customer hotline and have her sign up herself, maybe she will save more money
c. He should show her all the cell phone plans offered at the store and have her decide on the best deal
d. He should take her information down, contact TeleSERVE USA to set up her service, and sell her the TeleSERVE USA

Cheryl loves to buy greeting cards because she never knows when the time will come when she will need to give one to someone. She wants to develop a new greeting card filing system, because her collection of cards dramatically increased when the greeting card store had a going out of business sale. She has greeting cards for birthdays, encouragement, bereavement, thank-you's, Christmas, Easter, Mother's Day, Father's Day, you name it she has a greeting card for it. Some of the cards are blank so that she can put in her own message and others come with a printed message. They are in all different shapes, colors, and sizes.

19. How should Cheryl organize the greeting cards?

a. Order them by event, then by size, and then by blank or printed message inside
b. Order them by shape, then by event, and then by blank or printed message
c. Order them by size, then by shape, and then by event
d. Order them by blank or printed message inside, then by size, then by shape

Larry works as a manager in SuperBuys shoe store. SuperBuys shoe store is a major distributor of hard to find shoes in all sorts of colors and sizes. SuperBuys shoe store's supplier can find even the most hard to find items. Recently a customer came in and requested a pair of shoes that SuperBuy sold out of two months ago.

20. How should Larry handle the customer's request?

a. Take the customer's information, call his supplier to locate the shoes, and have the shoes shipped to the store
b. Explain that they sold out of the shoe two months ago
c. Take the customer's information, locate the shoes, and have the shoes shipped to the customer
d. Give the customer the supplier's information to contact the supplier directly

Wanda wants to develop a new system for handling committee proposal reviews; she has also hired a new office manager to implement the system. The proposals come from various business units within her multi-million dollar foundation. Each proposal takes up to two hours to read – depending on the business unit and the size of the project, one hour to write comments, and an additional thirty minutes to update her records on the work she has done with each proposal. The proposal review system should hold information about the proposal (authors, date, and title), amount of time it took to review it, and approval/rejection status.

21. How should Wanda tell the office manager to set up the new protocol for proposals?

a. Design a system for each business unit, since they require differing amounts of time for proposal review
b. Design a system that takes general proposal information that can be tailored for each business unit
c. Design a system that based on the amount of time it takes to review a proposal
d. Design a system based on the approval/rejection status

Miguel is in charge of changing the bus route for the 2003/2004 school year. Due to recent budget cuts the bus company has to eliminate two routes. Miguel is an experienced driver of twelve years and has working knowledge of all the existing routes. There are currently ten routes, with buses leaving every fifteen minutes between the hours of 6:00 a.m. and 8:00 p.m. and every thirty minutes between the hours of 8:01 p.m. and midnight. Typically it takes one bus thirty minutes to complete an entire route loop.

22. How should Miguel go about changing the new routes?

a. Start with the existing route and modify it based on suggestions from the drivers
b. Modify some of the existing routes to include the stops of the two of the least frequently used routes.
c. Eliminate the least frequently used routes and increase the frequency of other bus routes.
d. Redesign all the bus routes to cover more territory and to make more stops

Denise wants to teach her younger brother the difference in a bird and a duck. She is having tough time telling them apart because they both have feathers and a beak. It is approaching winter and there are birds and ducks gathered around a nearby pond. Denise decides to walk her brother to the pond to explain to him the difference in the creatures.

23. What characteristics should she point out to explain the difference in a bird and a duck?

a. Ducks quack and have bills and birds honk and have beaks
b. All Birds fly south for the winter in a "V" shape and ducks usually swim during the spring/summer season and camp in grassy-wooded areas during the fall/winter season
c. Ducks are usually larger than a bird and have more feathers
d. A duck is a kind of bird that usually resides in or around water where most birds can fly and usually stay high up in things like trees and on roofs to avoid attack

Mary wants to pack all of her furniture before she leaves Anaheim to move to New Jersey. She has so many tasks to complete before she leaves and she has less than two weeks to get rid of everything. She has three bedroom sets, two dinette sets, two sofas, three big comfy chairs, four closets full of clothing, two closets full of shoes, and tons of wall hangings. She also needs to call and have new services (phone, electricity, water, gas, cable) connected in New Jersey and have her current services scheduled for disconnected.

24. How should Mary pack her things?

a. Group the tasks that must completed into common clusters, complete all tasks from in one cluster and then start the next cluster of tasks
b. Create a list of everything she has to pack and cross things off as she goes along
c. Pack the things that will take the most time and pack the smaller things as time permits
d. Pack items at night, based on a list, and call the service companies whenever time permits

Margaret is a second year computer science student at Hokie Institute. She just received a homework assignment that required her to develop a general algorithm for swapping two items in an array.

25. How should Margaret design the algorithm?

a. She should first think of possible data values that would be stored in the array and then develop rules for each data type.
b. She should solve the assignment as if she were swapping simple integers and change the data type as necessary
c. She should solve the assignment as if she were swapping a specific item type then change the item type to some arbitrary value that's easily replaceable
d. She should think about how to swap items first and then think about the data type

David works as a statistician in a major law firm in New York City. The top lawyer of the firm came to David on yesterday and asked for figures on the number of drunk-driving fatalities in the New York City metropolitan area.

26. How should David present the information to the lawyer?
    a. Give a table showing the analysis results with a descriptive summary of table
    b. Give him tables, numbers, and instructions on how to get the most information out of all the data
    c. Give in a detailed report containing pages of tables and facts from each table
    d. Give him raw data and show him how to compute the correlation of multiplicity, this way he can retrieve as much information as needed

Rick is in Las Vegas at the blackjack table. Rick and several of his friends have been playing for several hours now. It is time for the dealer to change decks of cards because the current cards are worn. They don't want to take a break because they are starting to win.

27. How should the dealer change the deck without interrupting the game?
    a. Ask Rick and his friends to step away from the table
    b. Have the new deck on stand-by and after the current hand is complete, switch the decks
    c. While they are looking at there current hand, switch the remaining cards out for the new deck of cards
    d. Ask Rick and his friends to place their wagers, as they are deciding on their wagers, switch the decks of cards

Daniel wants to open his restaurant in Illinois selling pita pockets. He has done a lot of research on vendors and supply chains. Daniel wants a specific type of pita bread only sold in California, but according to local laws he must buy products from central Illinois. There is one baker that has rights to import goods from California and sell them in Illinois. Daniel contacted this baker, only to find out that the baker just sold his last shipment of rare pita bread.

28. How should baker handle Daniel's request?
    a. Have Daniel contact the California vendor directly and ship the pita bread to the licensed baker in Illinois
    b. The baker should contact the pita guy and order extra pita for Daniel on the next shipment
    c. The baker should explain his ordering process, showing him the details of how he orders, so that when Daniel opens his business he will already know the process
    d. The baker should explain to Daniel that he is out of bread, but can get more special pita bread as requested.

APPENDIX E

PRE/POST TRAINING DESIGN TASK

THE ART OF DESIGN

DESIGN TASK

You have been hired as an Expert Game Designer for ACME Gaming Inc. As a game designer, you are the visionary person on a team consisting of a software designer, developer, and tester. You are not responsible for coding, but the software developer will create the games according to your design specifications. As your first design project ACME Gaming Inc. wants you to come up with the design specifications for a new and exciting **TERRARIUM** game.

You are responsible for designing a terrarium game using the design notation as shown in the attached design example. While performing this task, please verbally communicate your thoughts, feelings, and ideas about your design process. So, as you are performing a task, begin the task, begin the task by saying something on the lines of "I am about to start drawing…", "I think that this solution will work, but it needs…", "Wait! That leads to another issue…", etc.

Read the following overview on the Terrarium game. If you need clarification on anything, please ask before the actual design process begins.

This package includes:

- **An overview of the Terrarium Game**
- **A terminology reference guide** (terms included in this guide are underlined throughout this package).
- **A Design Example** (ATM machine)
- **Note Cards create components of the system**
- **Blank Paper to draw the overall picture of the system**
- **Tape**
- **Pencils**
- **Pens**

****

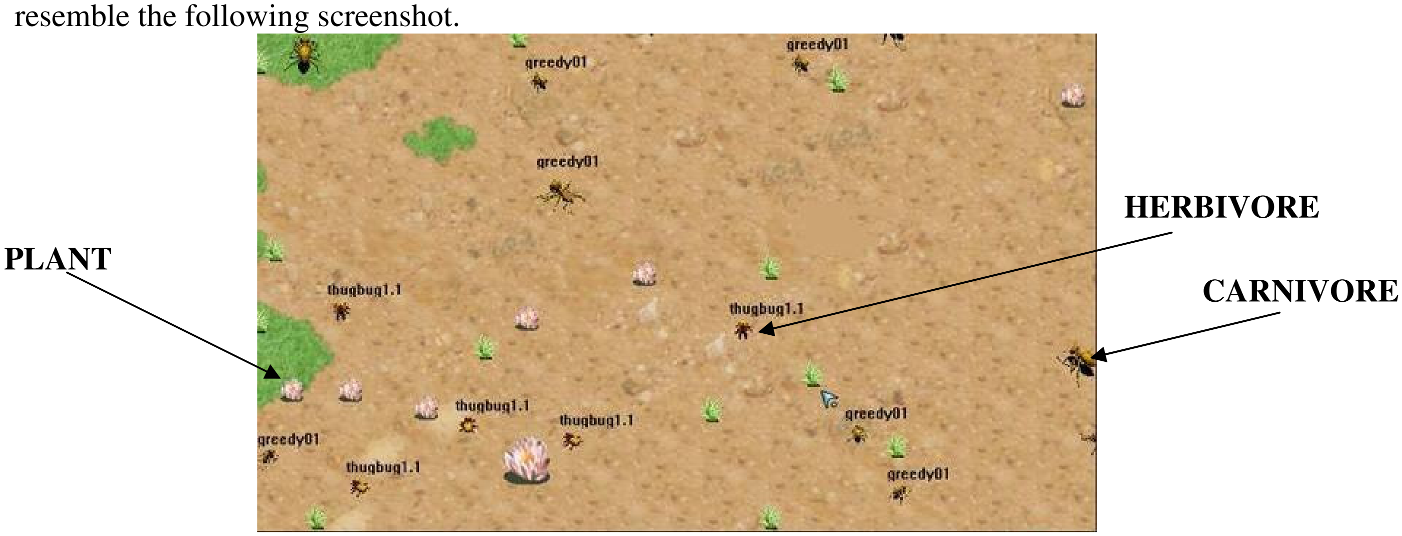
After you have read the overview, think about the design requirements. Have you seen/created anything similar to this in the past? Are there features of this design that remind you of something you've played before? Can you recall any concepts in previous computer science courses that will help you with this design creation?

****

THE TERRARIUM GAME

In Terrarium, players create creatures - herbivores, carnivores, or plants - and then introduce them into a networked ecosystem for a survival-of-the-fittest type competition. In creating a creature, players have complete control over everything from genetic traits (eyesight, speed, defensive power, attacking power, and any other exciting feature you can think of) to behavior (rules for locating prey, moving, attacking, etc.) to reproduction (how often a creature will give birth and which genetic traits, if any, will be passed on to its offspring). When a creature is initially introduced into the ecosystem, ten instances of it are scattered throughout the ecosystem. Creatures may die from old age, starvation, or predator attack. No more instances of that creature may be introduced until the creature has died off completely. However, based on the reproduction rules the creature may reproduce, thus perpetuating its lifecycle. The Terrarium game should provide a competitive medium for testing a player's strategic thinking skills as their creatures fight for survival in an unknown environment. The ultimate goal for a player is for their creature to be the last "man" standing.

After you have completed your design, the developers will create a gaming environment that will resemble the following screenshot.



While creating about the layout of your design, think about the following questions:

- How many components will the overall system have?
- What are the names of each of the components?
- What are the attributes and/or responsibilities of each component?
- How will the components work together?
- Are there components that could be grouped into a larger component?
    - If so, what similarities in the components caused you to form this larger component?
- Are there component attributes that everyone playing that game should be able to view (public attributes); similarly, are there component attributes that should be kept private and out of view from others (private attributes)?
- Are there any actions or tasks each component should perform?
- Is there a single component that can contain all other components?
- How will the components communicate (ex. how will something know it was eaten or how will something know it ate something else?)

TERMINOLOGY REFERENCE GUIDE

1. **Attribute** – A quality or characteristic inherent in or ascribed to someone or something.

2. **Behavior** – The action or reaction of something under specified circumstances.

3. **Carnivore** – Any flesh-eating animal.

4. **Component** – A part or element of an overall system.

5. **Creatures** – Creatures are considered to be herbivores, carnivores, or plants.

6. **Design** – A graphic representation of a detailed plan for construction of a system. A basic scheme or pattern that affects and controls function or development.

7. **Designer** – One who specializes in the construction of detailed plans of a system.

8. **Design Notation** – A system of phrases and figures used within design to symbolize components. Each component should have a name, responsibilities, and possible collaborators.

9. **Developer** – One who specializes in developing source for systems based on design documents created by designers.

10. **Ecosystem** – a system formed by the interaction of a community of organisms with their environment.

11. **Genetic Traits** – Characteristics specific to a creature at the time of its creation. These characteristics may be passed on to its offspring.

12. **Herbivore** – An animal that feeds chiefly on plants.

13. **Networked Ecosystem** – An ecosystem that is controlled of a network. Where there many be multiple players within the same environment.

14. **Plant** – A vegetable; an organized living being, generally without feeling and voluntary motion, and having, when complete, a root, stem, and leaves.

15. **Responsibilities** – Tasks or behaviors a component must perform within the system.

16. **Terrarium** – A small enclosure or closed container in which selected living plants and sometimes small land animals are kept and observed.

17. **Tester** – One who specializes in assuring the correctness and robustness of a software development product.

DESIGN EXAMPLE

COMPONENT LAYOUT GUIDE

NAME OF "THING"	
RESPONSIBILITY	**COLLABORATORS**
{anything the "THING" knows about itself or any actions it can perform with the knowledge it has}	{any "THING" is used to get information for or perform actions for this "THING"}

The cards on the following page represent one possible design of an Automatic Teller Machine (ATM). The ATM machine is designed from the perspective of the customer and the operator. The operator can perform tasks related to setting the cash amounts, setting cash withdrawal limits, and checking the connection of the ATM to the bank's network. The customer can insert her/his card, perform a transaction (withdrawal, deposit, transfer, inquiry) and print a receipt.

****
IF YOU RUN OUT OF SPACE ON THE FRONT OF THE CARD, CONTINUE ON THE BACK
****

RECEIPT

RESPONSIBILITY	COLLABORATOR
Heading	RECEIPT PRINTER
Details	
Balance	

RECEIPT PRINTER

RESPONSIBILITY	COLLABORATOR
	RECEIPT ATM

ENVELOPE ACCEPTOR

RESPONSIBILITY	COLLABORATOR
acceptEnvelope	ATM

ATM

RESPONSIBILITY	COLLABORATOR
Run	RECEIPT PRINTER
CardInserted	CASH DISPENSER
GetCardReader	CUSTOMER CONSOLE OPERATOR PANEL ENVELOPE ACCEPTOR

SESSION

RESPONSIBILITY	COLLABORATOR
pin	TRANSACTION
display	CUSTOMER CONSOLE
	OPERATOR CONSOLE

CASH DISPENSER

RESPONSIBILITY	COLLABORATOR
cashOnHand	ATM
checkCashOnHand	MONEY
dispenseCash	

TRANSACTIONS

RESPONSIBILITY	COLLABORATOR
balance	ATM
pinCheck, performTrans	ACCT TRANSACN
transactMenu	

CUSTOMER CONSOLE

RESPONSIBILITY	COLLABORATOR
DisplayMenu	ATM
readPIN readMenuChoice	

OPERATOR CONSOLE

RESPONSIBILITY	COLLABORATOR
setInitialAvailCash	ATM
setWithdrawalLimit	MONEY

MONEY

RESPONSIBILITY	COLLABORATOR
dollarAmt	CASH DISPENSER
currentAvail	OPER CONSOLE

ACCT INFORMATN

RESPONSIBILITY	COLLABORATOR
acctNum	WITHDRAWAL
getBalance	DESPOSIT, INQUIRY
setBalance	TRANSFER

CARD READER

RESPONSIBILITY	COLLABORATOR
readCard	ATM
ejectCard	CARD
retainCard	

CARD

RESPONSIBILITY	COLLABORATOR
getNumber	ATM
	CARD READER

WITHDRAWAL

RESPONSIBILITY	COLLABORATOR
fromAcct	ACCT INFORMTN
WithdrawFunds	

DEPOSIT

RESPONSIBILITY	COLLABORATOR
fromAcct	ACCT INFORMTN
DepositFunds	

TRANSFER

RESPONSIBILITY	COLLABORATOR
fromAcct	ACCT INFORMTN
TransferFunds	

INQUIRY

RESPONSIBILITY	COLLABORATOR
fromAcct	ACCT INFORMTN
Inquiry	

APPENDIX F

DESIGN TASK GRADING RUBRIC

Developed by Custer, Valessey, and Burke (2001)

	(1)	(2)	(3)	(4)	(5)
Examine context & define problem (COHESION)	▪ Tends to hone in on wrong problems. ▪ Doesn't see problem context. ▪ Most classes have operations that do not work together to support a single purpose.	▪ Tends to hone is on the easiest part to solve. ▪ Ignores problem context ▪ Several classes have operations that do not work together to support a single purpose.	▪ Tends to hone is on isolated subsets or the overall problem. ▪ May ignore problem context ▪ There are a few classes that have operations that work together, but do not support a single purpose.	▪ Identifies sub-problems but does not prioritize. ▪ Explores areas of the problem context. ▪ Most class operations logically fit together to support a single, coherent purpose.	▪ Identifies and prioritizes sub- problems (within the larger problem). ▪ Explores problem context. ▪ All class operations logically fit together to support a single, coherent purpose.
Generate & visualize possible classes (CONTENT)	▪ Cannot identify classes or the identified classes are inappropriate to framed problem. ▪ Does not appear to have an idea of where to begin ▪ Classes are disconnected from, or totally ignore, constraints.	▪ Identifies classes that meet some of the constraints. ▪ Some of the classes are adequate to solve the problem. ▪ Classes may/may not be feasible.	▪ Generates classes that meet most of the constraints. ▪ Generates several possible classes that are quite feasible, but misses some system constraints. ▪ Thinks "inside the box".	▪ Generates feasible classes. ▪ Sufficient number of classes to cover all the system constraints. ▪ Proposes creative classes, but still thinks "inside the box".	▪ Generates creative and efficient classes. ▪ All classes meet overall system constraints and address the original problem. ▪ Think innovatively.
Select a design solution (COMPLETENESS)	▪ Selects solution according to personal preferences. ▪ Unable to decide on a design solution. ▪ Solution represents an easy way out.	▪ Select solution with limited attention to the initial criteria. ▪ Solution may or may or be feasible. ▪ Is tentative and insecure in the selection of class operations to support the overall design solution	▪ Selects a reasonable solution based on criteria. ▪ Solution meets overall system constraints. ▪ Most class operations support the overall design solution.	▪ Selects solution on basis of efficiency and effectiveness. ▪ Checks against the initial system constraints. ▪ Provides rationale for selection.	▪ Provides detailed reasons for selecting solution. ▪ Briefly discusses an alternate solution ▪ Attempts to be innovative and wants the best solution possible.
Plan & communicate design (CLARITY)	▪ Explains design in general terms with little detail ▪ Sketches are rough and without sufficient detail. ▪ Ignores all constraints.	▪ Explains design plan, citing procedures, and other requirements. ▪ Visualizes using technical sketches without regard for scale. ▪ Ignores key constraints	▪ Creates an organized plan with sufficient detail. ▪ Visualizes using technical drawings. ▪ There is little confusion about there ability to adhere to the system constraints.	▪ Explains design ideas with detail ▪ Creates a plan with supporting technical drawings. ▪ Meets system constraints.	▪ Develops detailed design plan, drawings, and sketches. ▪ Devotes careful attention to constraints. ▪ Continuously revisits constraints and refines solution accordingly.
Test & critique solution (CONSISTENCY)	▪ The solution fails to meet the system constraints and design problem. ▪ In spite of problems detected, no effort is made to refine the solution. ▪ No subsequent drawings or design documentation.	▪ The solution is only marginally connected to the design problem. ▪ Shows little interest in improving the solution. ▪ Drawing/documentation is minimal	▪ The solution addresses some aspects of the design problem, but ignores others. ▪ Recognizes the need for improvement. Some ideas are generated, however only in concept. ▪ Drawing/documentation is sketchy	▪ The solution meets most of the constraints and criteria. ▪ Some general improvement ideas are generated and documented. ▪ Drawings are reasonably understandable.	▪ The solution fully consistent with the design constraints and criteria. ▪ Specific improvement ideas are generated and documented. ▪ Drawings are understandable.
Refine solution (CORRECTNESS)	▪ Solution is accepted "as is". ▪ Criteria and constraints are not referenced. ▪ Total disregard for correctness of overall solution.	▪ Some minor refinement of the original solution. ▪ Refinements are primarily cosmetic in nature and contribute only marginally to the quality or effectives of the solution. ▪ Little regard for correctness of overall solution.	▪ Solution is refined to be consistent with design constraints and criteria. ▪ Changes represent some improvement to the quality and functionality of the solution. ▪ Verbal account of correctness of solution, but actual solution does not reflect correctness plans.	▪ Solution is refined in a manner consistent with almost all the constraints and criteria. ▪ Changes represent some improvement to the quality of the solution. ▪ Solution partially reflects correctness.	▪ Solution is refined in a manner consistent with constraints and criteria. ▪ Changes represent substantial improvement to the quality of the solution. ▪ Solution totally reflects correctness.

APPENDIX G

CURRICULM VITA

TRACY L. LEWIS

207 Chowning Place
Blacksburg, VA 24060
(540) 961-1244
tracyL@vt.edu

EDUCATION

PH.D. COMPUTER SCIENCE
Virginia Polytechnic Institute and State University, Blacksburg, VA
Summer 2004.
Dissertation Title: Design Readiness: An Exploratory Model of Object-Oriented Design Performance
Advisors: Drs. Mary Beth Rosson & Manuel Pérez-Quiñones

M.S. COMPUTER SCIENCE – EMPHASIS IN HUMAN COMPUTER INTERACTION
Virginia Polytechnic Institute and State University, Blacksburg, VA
December 2001.

M.S. COMPUTER SCIENCE – SOFTWARE ENGINEERING
North Carolina Agricultural and Technical State University, Greensboro, NC
December 1998.
Thesis Title: The Diffusion of Technology as it Applies to Software Pattern Recognition
Advisor: Dr. David Bellin

B.S. COMPUTER SCIENCE
Tuskegee University, Tuskegee, AL
May 1996.

EXPERIENCE

February 2004 – Present
Assistant Professor — **Radford University**
Department of Information Science and Technology — **Radford, VA**
Principles of Computer Science I
- Course Instructor for one section of the foundation course that provides a rigorous, systematic approach to problem solving and programming.

Principles of Computer Science II
- Course Instructor for two sections the development of a disciplined approach to programming, with emphasis on data abstraction.

Computer Science Education Research
- Conduct studies that used to identify predictive experiences in object-oriented programming.

August 2003 – May 2004
Graduate Teaching Assistant — **Virginia Tech**
Department of Computer Science — **Blacksburg, VA**
Introduction to Object-Oriented Programming – Java
- Lab Instructor for three sections of the first computer science course for incoming freshmen.
- Implemented the principles of pair-programming in a closed lab environment – where two students worked side-by-side (as navigator and driver) at one computer, collaborating on design, algorithms, coding, and testing.

Introduction to Object-Oriented Design – Java

- Lab Instructor for three sections of the second computer science course for incoming freshmen.
- Continued educating students on the fundamental principles of object-oriented programming. Techniques of documenting object-oriented design trade-offs were introduced.

January 2000 – August 2003
Graduate Research Assistant **Virginia Tech**
Center for Human Computer Interaction **Blacksburg, VA**

- Software/Usability Engineer of visual programming environment to support novice programmer development of educational simulations.
- Development of patterns-based tutorial materials to effectively teach programming to novices using a visual programming environment.

December 1997 – October 2001
Principal Research Investigator **Bell Laboratories**
Lucent Technologies
Software Production Research Department **Naperville, IL**

- Lead research efforts to assess the education and dissemination of software patterns within organizations.
- Served as one of the department experts on the application of Software Design Patterns.
- Participated on a research team, sponsored by the National Science Foundation, which analyzed the organizational aspects of "code decay" on coding subsystems.

June 1996 – December 1997
Member Of Technical Staff (MTS) **Lucent Technologies**
Operator Services Position System (OSPS) **Naperville, IL**

- Responsible for analysis, design, coding, and testing of assigned Operator Services features specified for the 5ESS switch and its peripheral equipment.
- Served as a Technical Recruiter for Historically Black Colleges and Universities.

Summer 1995
Senior Technical Associate (STA) **AT&T**
Customs Manufacturing Services (CMS) **Greensboro, NC**

- Initiated and implemented a database to maintain hardware and software inventory.
- Developed a graphical user interface for users to access data stored in the database.

Summer 1994
Junior Systems Analyst **Commonwealth Edison Power Co.**
Information Systems **Chicago, IL**

- Designed a statistical method of detecting equipment failure in the earliest possible stages.

TECHNICAL EXPERTISE

Computer Skills

Application Programming, System Installation/Monitoring/Coordination, Web Page Design, Database Administration, Software Implementation and Documentation.

Courses

Telecommunications, Human Factors Research Design, HCI for Collaborative Systems, Computer Supported Collaborative Works, Object Oriented Analysis/Design, Models of HCI, World Wide Web Programming, Digital Libraries, Information Storage and Retrieval, Project Management, Software Analysis and Design, C/C++ Programming, Java, UNIX, Programming Languages, Operating Systems, E-Commerce

Software

Visual Programming Environments, Rational Rose, Visual Studio .NET, Smalltalk, Microsoft Access/Visual Basic, Authorware Professional, Dreamweaver, BlueJ, Eclipse

Hardware

DEC, NT, Alpha, and SUN Workstations. IBM, IBM compatibles, and Macintosh

Languages

C, C++, Java, HTML, Smalltalk, Scheme, Lisp, Prolog, Pascal, COBOL, FORTRAN, LaTex, Perl, UNIX shell scripts

REFEREED CONFERENCE PAPERS

- Lewis, T.L., Perez-Quinones, M, Rosson, M.B. (accepted March 2003). **A Comprehensive Analysis of Object-Oriented Design: Towards a Measure of Design Readiness.** Frontiers in Education (FIE) 2004.
- Lewis, T.L., Rosson, M.B., Perez-Quinones, M. (March 2003). **What Do the Experts Say? Teaching Introductory Object-Oriented Design From an Expert's Perspective.** ACM Special Interest Group on Computer Science Education (SIGCSE) 2004.
- Lewis, T. L., Rosson, M. B., Carroll, J. M., Seals, C. D. (Sept 2002). **A Community Learns Design: Towards a Pattern Language for Novice Visual Programmers.** IEEE Human Centric Computing Languages and Environments HCC02.
- Rosson, M. B., Carroll, J. M., Seals, C. D., Lewis, T. L. (July 2002) **Community Design of Community Simulations.** IEEE Designing Interactive Systems DIS2002.
- Seals, C. D., Rosson, M. B., Carroll, J. M., Lewis, T. L. (March 2002). **Fun Learning Stagecast Creator: An exercise in Minimalism and Collaboration**. IEEE Human Centric Computing Languages and Environments HCC02.

BOOK CHAPTERS & MAGAZINE ARTICLES

- Lewis, T. When **Are Students Ready to Learn Design? A Measure of Design Readiness.** ACM CrossRoads Magazine – Special Edition on Computer Science Education (Submitted For Review – Nov 2003).
- Doswell, F., Harley, H., Lewis, T., Seals, C., and Dr. G. Scales. (Feb 2002). **Adapting to Life as a Graduate Student: Getting Up to Speed on Information Technology**. A Black Student's Guide to Graduate and Professional School Success, Published Jan 2003.

PRESENTATIONS

- Object Oriented Programming, Systems, Languages, and Applications Conference (Nov 2002). Doctoral Symposium - **A Measure of Design Readiness: Using Patterns to Facilitate Teaching Object-Oriented Design.**
- European Conference on Object-Oriented Programming (June 2002). Educator's Symposium - **A Measuring Design Readiness**.
- European Conference on Object-Oriented Programming – Malaga, Spain (June 2002). Doctoral Consortium - **Are Undergraduates Ready for Design?**
- Special Interest Group on Computer Science Education (Feb 2002). Doctoral Consortium – **Using Patterns to Teach Object-Oriented Design**.
- Association for Computing Machinery – Virginia Tech Chapter (Oct 2001). Presentation – **An Introduction to Object-Oriented Design Patterns.**

POSTERS

- Lewis, T. (March 2003). **What's Do the Experts Say? Teaching Introductory Design from an Expert's Perspective.** Virginia Tech Graduate Research Symposium.
- Lewis, T. (Nov 2002). **A Measure of Design Readiness: Using Patterns to Facilitate Teaching Object-Oriented Design.** Object Oriented Programming, Systems, Languages, and Applications Conference (OOPSLA).
- Rosson, M.B., Carroll, J.M., Seals, C. D., Lewis, T. L., Gornter, J., Snook, J. and Trolley T. (July 2002). **Community Design of Community Simulations.** IEEE Designing Interactive Systems DIS2002.
- Lewis, T. (March 2001). **Using Patterns to Teach Object-Oriented Design.** Virginia Tech Graduate Research Symposium.

HONORS AND AWARDS

- Virginia Tech Graduate Research Symposium Finalist, March 2003 – 3rd Place
- Minority Graduates in Engineering (GEM) Ph.D. Fellow, August 2000 – August 2001
- National Science Foundation Graduate Research Trainee at Virginia Tech, January 1999 – August 2000
- Minority Graduates in Engineering (GEM) Master's Fellow, August 1997 – December 1998
- Microsoft Scholar, August 1995 – May 1996
- AT&T Educational Scholar, August 1992 – August 1995

SERVICE AND PROFESSIONAL DEVELOPMENT

- Certificate of Completion – National Institutes of Health: Human Participants Protection Education for Research Teams – December 2003.
- Committee on the Status of Women in Computing Research Mentoring Workshop – February 2003.
- E-Commerce and Networking in the European Union. Study Abroad Program – Freiburg, Germany. Summer 2002.
- ACM International Student Poster Competition - Reviewer - February, 2001, 2002
- Computer Science Graduate Council Officer – Secretary of the Interior, Member of Graduate Program Committee, and Graduate Student Association Department Representative.