Information Technology and Management 6, 333–350, 2005

Cognitive Differences Between Procedural Programming and Object Oriented Programming

GARRY WHITE gw06@txstate.edu
MARCOS SIVITANIDES ms06@txstate.edu
Computer Information Systems, Texas State University at San Marcos, 600 University Dr., San Marcos, TX 78666

Abstract. Software development is moving from procedural programming towards object-oriented programming (OOP). Past studies in cognitive aspects of programming have focused primarily on procedural programming languages. Object-oriented programming is a new paradigm for computing. Industry is finding that programmers are having difficulty shifting to this new programming paradigm.

Findings in prior research revealed that procedural programming requires Piaget's formal operation cognitive level. New from this research is that OOP also requires Piaget's formal operation cognitive level. Also new is that OOP appears to be unrelated to hemispheric cognitive style. OOP appears to be hemispheric style friendly, while procedural programming is preferential to left hemispheric cognitive style.

The conclusion is that cognitive requirements are not the cause for the difficulty in shifting from procedural to OOP. An alternative possibility to the difficulty is proactive interference of learning procedural programming prior to learning object oriented programming.

Keywords: procedural programming, object oriented programming, cognitive style, cognitive development

Introduction

The trend in computer programming is toward the use of OOP languages [6,14,17,38], such as Java [22] and away from Procedural programming languages, such as COBOL. Even the newest COBOL compilers by MERANT/Micro Focus are object-oriented [43] thus supporting this trend. While legacy COBOL applications are by default procedural in nature, new COBOL applications are object-oriented in nature [13,51]. This trend is due to the demand for business programming applications that are designed for e-commerce through the Internet. Java is a programming language designed for Web application development on the Internet and it is also the open-ended interface to storage area networks.

OOP languages, such as Java, group data and procedures under one identifier, referred to as an object. Objects typically posses operations that change their state and can in turn invoke operations on other objects [24]. Designers map objects to real-world domain entities like teams, scores, and so forth [41]. Procedural Programming has data and procedures separate, with the instructions manipulating data. These procedures are sequences of statements that transform inputs to outputs [24]. Procedural designers rely on a kernel concept that effectively defines a top to the design [41].

Research suggests that the shift from procedural programming to OOP is difficult for programmers [17,34,49]. Yet, with this difficulty of learning OOP, there is research suggesting that object oriented properties are present in user cognition [28]. Some of the difficulties Sheetz (1997) found were the reuse of components and class library instead of creating beginning/end routines. Another difficulty was the usage of methods and message passing [49]. Pennington (1995) and Marshall (1995) found a tendency by programmers to implement procedural methods in an OOP environment rather than think in terms of object-oriented concepts. Procedural programmers must re-think software systems from the standpoint of reusable components rather than producing lines of new code for each application [5].

Research has shown that cognitive development/abilities (what can be learned), cognitive styles (how one learns), hemispheric brain dominance, and prior experiences are factors in learning procedural programming languages [11,18,31,32,36,39]. Specifically, Monfort et al. [36] found that a large number of disciplines (such as Accounting, Mathematics, Management, Nursing, Criminal Justice and Computer Science) attract left brain dominant students, while an equally large set of other disciplines (such as Advertising, Music, Art) attract right brain dominant students. Their study suggests that most students majoring in a specific discipline would be expected to be of the same hemispheric type. The lever of analysis of their study was the students and their major. However, even within a major, there are students who perform better in certain types of courses than others. So, based on their findings, our study reduces further the unit of analysis by studying students who are in the same major but have to deal with two different requirements within the same discipline of computer programming, that is, procedural and object oriented programming. Therefore, we expand on the Monfort et al. [36] study by investigating hemispheric brain characteristics within a highly specialized area in the discipline on computer programming. There has been very little research on the cognitive characteristics of the OOP methods compared to procedural programming methods.

This study investigated the required cognitive hemispheric dominance of the brain as defined by the Hemispheric Mode Indicator and cognitive development as defined by Piaget, for the OOP language Java. It is similar to a prior study using the OOP language C++ [52]. Relationships between cognitive requirements of computer programming languages and programmers' cognitive characteristics is the published theory supporting this research [53]. The three research questions were:

1. Do test grades in the Java OOP language correlate with cognitive development, as defined by Piaget and measured by the Propositional Logic Test (PLT)?
2. Do test grades in the Java OOP language correlate with cognitive hemispheric dominance, as measured by the Hemispheric Mode Indicator (HMI) inventory?
3. How do these cognitive characteristics for Java OOP compare with the cognitive characteristics for procedural programming?

Literature review

There are two areas of predictors on which this study will focus. These are (1) cognitive development and (2) cognitive hemispheric dominance/style [53]. It needs to be noted that the impact of these cognitive factors can vary in strength due to differences in course content. For example, a programming paradigm may involve concepts that favor the left side of the brain; another programming paradigm may involve concepts that favor the right side of the brain. The content for one programming paradigm may focus on object manipulation, while another may focus on problem solving skills and the flow of logic through the program.

Cognitive development component

There is a positive correlation between procedural programming and several cognitive abilities: spatial ability, general reasoning, analytic processing, and math skills [18]. Such relations have yet to be shown with OOP languages. Cafolla [11] showed a relationship between success in procedural programming, mathematics proficiency and Piaget's cognitive development.

Piaget's cognitive development theory deals with three levels of development [16,42], pre-operational, concrete, and formal operations. Pre-operational cognitive level is a very low level of thinking. Such a person can use symbols from visual and body sensation to represent objects but has problems mentally reversing actions [8]. An example of pre-operational is stating that a tall container has more water then a squat container, even if the person sees the water in the squat container is poured into the tall container. The next level, concrete, finds a person who can understand conservation of matter and classification/generalization (conclude that all dogs are animals and not all animals are dogs). However, such a person is unable to comprehend mathematical ratios (Barker, 1983). Formal operational level is the highest cognitive development level defined by Piaget. It is the ability to deal with abstractions, form hypotheses, solve problems systematically, and engage in mental manipulations [8]. Piaget's theory indicates that formal operational thinking abilities develop around age 12 [12]. It is at this age that some students begin to move from concrete thinking to logic/abstract thinking. Formal operational thinking is the highest cognitive level a person can achieve [8].

Several studies have shown a relationship between Piaget's formal operational thinking and success in procedural programming. Azzedine (1987) tested 203 students from the 6th grade to college level with the Langeot Test of Cognitive Development. That research investigated the implications of Piaget's cognitive developmental theory and the intellectual prerequisite of learning procedural programming. The results showed cognitive development predicted programming performance.

Cafolla [11] did a similar study with students from a community college. Each subject was given the Inventory of Piaget's Development Tasks (IPDT) to measure his or her cognitive development level. The results were the same as for Azzedine (1987): cognitive development predicted procedural programming performance.

The use of logic in computer science has been identified [20,21]. The ability to do Propositional Logic is part of Piaget's Formal Operational stage [9,15]. Propositional Logic has a direct analogy in procedural programming [19]. Propositional Logic involves truth tables. In procedural programming such concepts are found in IF/THEN/ELSE statements. The Propositional Logic Test (PLT) is an instrument that measures Piagetian tasks of logic.

Little [31] found that students who tested high in formal operation, a Piaget's high level of cognition, scored higher on procedural programming and logical thinking measures than those students who were concrete thinkers (a Piaget's lower level of cognition). This cognitive developmental level is a factor in determining one's ability to learn procedural programming [19]. This finding is also supported by Hudak and Anderson [26]. They determined that people, who have reached Piaget's formal operational stage, would have the tools needed to understand programming. They also have a greater abstract learning style that helps them learn programming.

The one study involving Piaget's formal operational stage and OOP was with C++ [52]. That study showed a relationship between C++ and formal operation thinking. This study does follow the same procedure and research method but uses Java as the OOP language being studied. Specifically, Java is used in a network of personal computers with a focus on developing Internet applications. Moreover, the subjects for this study are university sophomores and/or juniors. The white [52] study used Community College students as subjects and used C++ on a local PC environment to develop applications. So, while the white [52] study is used as a reference, its findings cannot be extrapolated to this study which albeit uses an OOP language but in a different operating environment and the subject populations are not similar. The findings of the white [52] study cannot be generalized (as with the findings of this study) to all OOP languages, under any development environment, for any subject. Therefore, repeated studies using similar OOP languages but different environments and subjects are warranted so that a body of theory can be solidified.

Likewise, a similar study suggesting that object oriented properties are present in user cognition [28] is also used as a reference. The study researched the relationship between object oriented properties and cognition. It did not study any possible relationships involving hemispheric style. Thus, since Krovi and Chandra [28] suggest findings between OO properties and cognition, this study expands on that by introducing an additional variable, the hemispheric style. It still studies cognition in order to confirm (or reject) the Krovi and Chandra [28] findings and to extend the research and the potential theory it introduces the additional variable of hemispheric style.

Hemispheric style component

Hemisphericity is a term used to describe how the brain processes specific information. Research suggests that one side of the brain predominates over the other [1,32]. The left brain functions differently from the right brain [10,50].

Students who are successful in procedural programming have been found to be significantly left hemispheric brain dominant for cognitive style. This was true at public, post-secondary and vocational-technical schools where "Your Style of Learning and Thinking-Form C" inventory forms were used [33]. A later study found Computer Science and Mathematics students also to be left brain dominant while music, art, oral communication and journalism students were found to be right brain dominant. Brain hemisphere dominance was inferred from Human Information Processing Survey scores [36].

A dissertation by Ott (1989) supports the above findings. She found that left-brain dominance in high school students, correlated significantly with grades in procedural programming courses ($r = .30$ & $.34$). Brain dominance was determined by the Herrmann Participant Survey Form and math scores of the Scholastic Aptitude Test (SAT-M) correlated much higher with procedural programming course grades ($r = .62$).

The one study involving hemispheric cognitive style and OOP was with C++ [52]. That study showed no relationship between C++ and hemispheric cognitive style. Right-brain thinkers do just as well as left-brain thinkers in C++.

Summary

The literature has shown that formal operational cognitive development is a required characteristic of people for learning procedural programming. Research has also supported another required characteristic for learning procedural programming. It is logical thinking skills, a component of formal operational cognitive development. The literature has also shown left hemispheric thinking style of learners as another characteristic necessary for success with procedural programming. Research has yet to firmly establish what these characteristics are for all types of OOP languages.

Research design

The cognitive characteristics issues this research investigated were (i) cognitive development, defined by Piaget and measured by the Propositional Logic Test (PLT), and (ii)cognitive style, as measured by the Hemispheric Mode Indicator (HMI). Correlation relationships between these characteristics and test grades were evaluated using subjects who were IS majors enrolled in a Java course at a university. Demographic data (handedness, footness, eye dominance, prior math and programming courses) were included in Step-Wise Regression analysis of each test grade.

The research Null Hypotheses are as follows:

Ho1. Cognitive development, as measured by the Propositional Logic Test (PLT), is not significantly correlated with achievement as measured by object oriented programming test scores.

Ho2. Cognitive style, as measured by the Hemispheric Mode Indicator (HMI), is not significantly correlated with achievement as measured by object oriented programming test scores.

In an introductory programming course for Information Systems majors at the university level, students received instruction in the content of Java. Both procedural and object oriented concepts were taught. The first test (Test #1) involved procedural programming concepts. Some of the concepts covered were loops, evaluate expressions, procedural logic, output, and data types. The second test (Test #2) dealt with both procedural programming and OOP concepts. Part of the second test covered the understanding of procedural logic and algorithm results. The rest of the second test included writing code for OOP methods, classes, and objects with the use of arrays. The third test (Test#3) was totally OOP concepts. The first part of the third test involved the determination of output from OOP code. The rest of the third test involved writing code for a method that used arrays and passed arguments between objects.

Sample size

Four weeks after the completion of the Java course, the researcher attempted to locate 100 students from the course. A large size was desired so as to detect any weak relationships. On the first class day, the researcher visited classes that were successors to the Java course. Sixty-two previous Java students were located. Of these, only 36 volunteered to participate. Such a sample size will only detect profound relations.

Descriptive statistics of subjects

Of the sample of 36 subjects, 15 reported to be male, 20 were female, and one made a non-response. Thirty-three subjects reported having had at least two semesters of high school Algebra. Nineteen reported taking at least one semester of a high school programming course. For 26 of the subjects, this Java course was their first college programming course. Thirty-two subjects were aged 19 to 21 with only four subjects aged 22 to 27. One subject left all the demographic data blank. Another subject entered unusual numbers for the semesters of prior courses (21 semesters for high school programming courses and 20 semesters for prior college programming courses). The data items from these two subjects were entered as missing data. Of the 36 subjects, 34 had full and complete valid demographic data. All 36 subjects did provide PLT and HMI data.

Research methodology

A Step-Wise Regression Analysis and Pearson's Correlation Coefficients were performed using all data variables and interactions. The probability of F to enter a variable in the Step-Wise Regression Analysis was set at $\leq .05$. The F probability to remove a variable was $\geq .10$.

The skewness and kurtosis were evaluated for each variable. Skewness indicates if the data are grouped at one end of the range of scores. Kurtosis indicates if the data is spread out (heterogeneous) or narrow (homogeneous). If the variable had significant skewness or kurtosis, results may suggest differing interpretations.

Methodological assumptions

Extraneous variables for this study are: maturity of the student, attitude toward learning, motivation [48] and self-confidence. It is possible that a student may have high logic ability and yet have low motivation to perform on the PLT, the HMI, or in the course. Hence, his or her data may not truly show the ability to be successful in the course. Since participation was voluntary, it is assumed that these extraneous variables have a minimum impact or effect, and subjects put forth an effort to complete both the PLT and the HMI. One of the issues we contended with, was whether any prior programming experience that the subjects may had had could have an effect and become a control variable. During the research design we realized that it would be very difficult to control our study for the content of prior programming experience. The differences in the levels of type, quality and quantity of that experience vary so much that controlling for it would be virtually impossible. Our results, however, showed us that controlling for prior experience was not necessary. What we found superseded any prior experience, in that even though the subjects had different backgrounds and we were unable to control for those variables, we still found statistical differences in our results, meaning that those uncontrollable intervening variables lacked an impact on our study.

Instrumentation

The Propositional Logic Test, developed at Rutgers University, is a test that measures Boolean (true or false) logical thinking ability. It measures logical thinking by asking students to interpret truth function operations with a stated logic rule. Enyeart [15] found propositional logic related to Piaget's Formal Operational cognition. The PLT can indicate Piaget's cognitive development level. The Test contains 16 propositional logic items. The score range is from 0 to 16.

HMI scores

An instrument that measures cognitive hemispheric dominance is the Hemispheric Mode Indicator (HMI) by Excel, Inc [35]. Characteristics for Left/Right Hemispheric Modes include: rational vs. intuitive, logical vs. hunches, differences vs. similarities, and objective vs. subjective judgements. The HMI stresses the cognitive aspects of hemispheric dominance. It has been used to study learning styles.

Thirty-two choices are given to the subject who then selects how well the particular item describes himself or herself. A score between -60 to $+60$ is tabulated. A preference for left dominance has a score range of -60 to -8. A mixed/whole brain dominance has a score range of -8 to $+8$. A right brain dominance has a score range of $+8$ to $+60$ [30].

Data analysis

SPSS was used to perform data analysis for the programming language test scores. Descriptives were used to look at distributions, skewness, and kurtosis of PLT, HMI, and test scores. This approach was used to verify whether the distributions were normal or not. Pearson's Correlations were also performed on this data.

Four linear Step-Wise regressions were performed for each test grade and average grade for the Java course. Grades were the dependent variable. Physical dominance (based on handedness and footedness and eye dominance), prior math and programming courses, HMI, and PLT were the independent variables. Interactions were also included. Each of three Java course test grades and the average of the three tests were used in a Step-Wise regression. Weights from the regression equations indicated which variables have the greater impact on the course grades. A correlation matrix of variables was provided for the programming language and checked for significance.

Findings

PLT characteristics

The Propositional Logic Test (PLT) scores were skewed to the high end (-1.034 skew/ .393 std error < -2) and had a normal spread of scores ($-.363$ kurtosis/.768 std error > -2). The mean was 12.19.

HMI characteristics

The Hemispheric Mode Indicator (HMI) scores formed a normal distribution (.$-.079$ skew/.393 std error > -2; .634 kurtosis/.768 std error < 2). The mean was -3.08 with a standard deviation of 14.85. As seen in Table 1, 36.1% were left-brain. They had a score of -8 or less. Whole brain thinkers made up 38.9% of the sample. Their HMI scores were between -8 and $+8$. This indicates mixed dominance. The scores of $+8$ or greater were composed of right brain thinkers. They composed 25% of the sample.

Tests characteristics

The scores of the three tests given in the course correlated significantly with each other. The values for Pearson's r ranged from .388 to .805; p values ranged from .000 to .019 (See Table 2). This is interpreted to show consistency in the instructor's subjective grading. The comparison between the first test and the third test provided an opportunity to compare the cognitive characteristics between procedural programming and object oriented programming.

Table 1
Cognitive brain side frequencies as measured by the HMI.

	Frequency	Percentage (%)
Left	13	36
Whole	14	39
Right	9	25

Table 2
Correlation Matrix of HMI, PLT, and 3 tests.

		PLT	HMI	Phy Dominance	Test 1	Test 2	Test 3
PLT	Pearson Correlatior	1.000	−.236	−.137	.416*	.419*	.369*
	Sig. (2-tailed)	.	.165	.424	.012	.011	.027
	N	36	36	36	36	36	36
HMI	Pearson Correlatior	−.236	1.000	.128	−.393*	−0.47	−.092
	Sig. (2-tailed)	.165	.	.456	.018	.787	.595
	N	36	36	36	36	36	36
Phy Dominance	Pearson Correlatior	−.137	.128	1.000	−.324	−.045	−.187
	Sig. (2-tailed)	.424	.456	.	.054	.795	.275
	N	36	36	36	36	36	36
Test 1	Pearson Correlatior	.416*	−.393*	−.324	1.000	.388*	.469*
	Sig. (2-tailed)	.012	.018	.054	.	.019	.004
	N	36	36	36	36	36	36
Test 2	Pearson Correlatior	.419*	−.047	−.045	.388*	1.000	.805**
	Sig. (2-tailed)	.011	.787	.795	.019	.	.000
	N	36	36	36	36	36	36
Test 3	Pearson Correlatior	.369*	−.092	−.187	.469**	.805**	1.000
	Sig. (2-tailed)	.027	.595	.275	.004	.000	.
	N	36	36	36	36	36	36

*Correlation is significant at the 0.05 level (2-tailed).
**Correlation is significant at the 0.01 level (2-tailed).

Test #2 contained both procedural and object oriented items. This test had a normal spread of grades based on the kurtosis (−.184 kurtosis/.768 Std. Error > −2). Kurtosis measures how narrow or flat the score distribution is. Test #1 and Test #3 are described as having homogenous values due to a significantly narrow distribution of grades (4.578 kurtosis/.768 Std. Error > 2 for Test #1 and 2.298 kurtosis/.768 Std. Error > 2 for Test #3). This narrow distribution of grades may cause a restriction for correlation statistics with other variables that have a normal spread or a flat spread of values (A narrow distribution of scores limits the range of possible paired relationships).

Correlation overview

As shown in Table 2, the PLT ($p < .05$, two-tail) was significantly related to all three tests. The HMI ($p < .05$, two-tail) was only significant with Test #1, procedural concepts. Even with kurtosis restricting the potential correlation range, Test #1 was significant with the PLT and HMI. And Test #3 was also significant with the PLT.

All three tests were significantly related to each other as shown in Table 2. Student performance appears to be consistent across tests. It is important to note that Test #3, object oriented concepts, was significant at $p < .01$ with both Test #1 and Test #2. What is even more important to note is the relationship between Test #2 and Test #3 ($r = .805$, $p < .000$). The difficulty level and content may have been more similar between these two Tests than between Test #1 and Test #2 ($r = .388$, $p < .019$).

Table 3
List of variables used for Step-Wise Regression.

	N	Mean	Std.	Skewness		Kurtosis	
	Statistic	Statistic	Statistic	Statistic	Std. Error	Statistic	Std. Error
Test 1	36	93.58	7.78	−1.849	.393	4.578	.768
Test 2	36	88.50	11.92	−.965	.393	−.184	.768
Test 3	36	86.00	17.16	−1.666	.393	2.298	.768
HMI	36	−3.08	14.85	−.079	.393	−.634	.768
PLT	36	12.19	4.22	−1.034	.393	−.363	.768
Phy dominance	36	2.44	.69	−.874	.393	−.393	.768
HS algebra	35	2.63	1.24	.770	.398	.455	.778
HS prog	34	1.21	1.55	1.802	.403	3.662	.788
College prog	34	.29	.63	2.795	.403	9.600	.788
Tot prior courses	34	1.50	1.66	1.638	.403	3.508	.788
HMI & PLT	36	−52.0000	192.9346	−.698	.393	−.102	.768
HMI & PLT & PhyD	36	112.6389	479.3263	−.643	.393	−.034	.768
HMI & PhyDom	36	−6.2500	38.7404	−.010	.393	−.272	.768
PLT & PhyDom	36	29.4167	13.2458	−.140	.393	−1.273	.768
Valid *N* (listwise)	34						

Hypothesis #1. Cognitive development, as measured by the Propositional Logic Test (PLT), is not related with grades in object oriented programming

Cognitive development, as measured by the Propositional Logic Test (PLT), had a significant correlation with all test grades, at the .05 Alpha level (Table 2). There were significant correlations between Test #2, partly OOP ($r = .419$, $p < .011$), and Test #3, totally OOP ($r = .369$, $p < .027$). Pearson r squared accounted for a 14 to 16% portion of the variance. The Null Hypotheses #1 was rejected.

Hypothesis #2. Cognitive style, as measured by the Hemispheric Mode Indicator (HMI), is not related to grades in object oriented programming

Brain hemispheric dominance, as measured by the HMI inventory, did not correlate with tests containing object oriented concepts (i.e., Test #2 and Test #3). The Null Hypotheses #2 was not rejected.

Step-wise regression findings for the three dependent variables (GRADES)

This research provided the opportunity to compare factors between procedural programming operationalized as Test #1, and OOP operationalized as Test #3. Since the same students took both tests, the same programming language was used, and the same instructor evaluated the tests, intervening variables were believed to be at a minimum. Such comparisons can give educators better insight as to the differences between these two languages and to identify why there was difficulty for a procedural programmer to learn object oriented programming.

Step-Wise Regression was performed on the three dependent variables: Test #1, Test #2, and Test #3. Table 3 lists these variables and the independent variables used. The three Tests were combined into a fourth dependent variable, an overall average course grade for a final Step-Wise Analysis. Each Step-Wise Regression was run twice. The first regression analysis was with prior courses and the interactions of PLT, HMI, and physical dominance. The second regression analysis was without the interactions and prior courses. This was done in order to compare the influences of prior courses and the interactions.

Step-Wise Regression findings for Test #1, procedural programming concepts

Step-Wise Regression included only HMI and PLT in the calculations for Test #1, procedural concepts. When interactions and prior courses were used, the PLT and the interaction of HMI and Physical Dominance had the same absolute Beta weight values, −.330 and .330 ($R = .519$, $p < .008$). None of the prior high school and college course variables of Algebra and programming were entered into the regression equation.

When interactions and prior courses were deleted, the HMI and PLT provided a similar R value and a similar p value ($R = .515$, $p < .006$). Procedural concepts revealed a relationship to hemispheric dominance. However, the PLT absolute Beta weight (.342) was greater then the HMI absolute Beta weight (−.312). This suggests propositional logic is a stronger characteristic than cognitive hemispheric dominance.

Step-wise regression findings for Test #2, procedural programming and object oriented concepts

Test #2 Step-Wise Regression included only PLT ($R = .454$, $p < .007$) when interactions and prior courses were considered. None of the prior course variables, Algebra and programming courses, entered into the regression equation. When interactions and prior courses were deleted, the PLT again was the only variable used ($R = .419$, $p < .011$). This test covered both procedural and object oriented concepts. Unlike Test #1 that had only procedural concepts, HMI was excluded in this Step-Wise Regression.

Step-wise regression findings for Test #3, Object Oriented programming concpets

Test #3, like Test #2, used only the PLT ($R = .373$, $p < .030$) when interactions and prior courses were considered. None of the other variables entered into the regression equation. When interactions and prior courses were deleted, the PLT again was the only variable used ($R = .369$, $p < .027$). This suggests that object oriented concepts, like procedural concepts from Test #1, require propositional logical thinking. However, this analysis for Test#3 excluded the HMI.

Step-wise regression findings for overall test average for programming course

The step-wise regression loaded only the PLT for the average of the three tests ($R = .463$, $p < .006$) when interactions and prior courses were considered. None of the prior course variables entered into the regression equation. When interactions and prior courses were deleted, the PLT again was the only variable used ($R = .455$, $p < .005$). This suggests that the grade earned in a programming course containing both object oriented concepts and procedural concepts relate to the PLT. Again the HMI was excluded.

Summary of findings

The correlation matrix (Table 2) indicated that both procedural concepts (Test #1) and object oriented concepts (Test #3) correlated to the PLT. The correlation matrix (Table 4.2) also indicated the HMI related ($r = -.393$, $p < .018$) to procedural concepts (Test #1). The direction of the HMI was to the left cognitive hemisphere.

Step-wise regression analysis was performed on the data. The HMI was entered only into the procedural concepts (Test #1) step-wise regression. Other variables (physical dominance, prior high school and college Algebra, and programming courses; and interactions) were never entered into the step-wise regression. Neither did these variables generate a larger R value when compared with using only the three independent variables (PLT, HMI, Physical Dominance). Also these other excluded variables did not correlate with the three tests.

This study revealed that the PLT is related to object oriented concepts (rejection of Null Hypothesis #1). Formal operational cognitive level is a characteristic that affects learning OOP. The HMI was unrelated to object oriented concepts (acceptance of Null Hypothesis #2). Hemispheric dominance, cognitive style, does not affect learning OOP.

Discussion

This study investigated cognitive characteristics' effects on learning Java, an OOP language. The characteristics were (i) cognitive development, as defined by Piaget, and (ii) cognitive hemispheric style, as defined by the Hemispheric Mode Indicator. The findings are as follows:

Hypothesis #1. "Cognitive development, as measured by the Propositional Logic Test (PLT), is not significantly correlated with achievement as measured by grades in OOP." This hypothesis was rejected. That is, cognitive development affects success in learning OOP.

Hypothesis #2. "Cognitive style, as measured by the Hemispheric Mode Indicator (HMI), is not significantly correlated with achievement as measured by grades in OOP." This hypothesis was not rejected. That is, OOP is cognitive style friendly. A person can learn OOP regardless of his/her cognitive style.

The literature indicates that procedural programmers have obtained Piaget's formal operational cognitive level. This is consistent with the findings of Hypothesis #1. It appears that programming using OOP also involves Piaget's formal operational cognitive level as indicated by PLT.

The literature indicates that procedural programmers are left-brain dominant. Hypothesis #2 of this research indicates the contrary for OOP. The results from the step-wise regression analysis also support this finding. Successful OOP programmers are either right-brain or left-brain or whole brain dominant.

Strength of the statistics and design used in the investigation

Homogenous distribution of test grades
Based on Kurtosis, Test #1 and Test #3 had a significantly narrow distribution of grades (4.578 Kurtosis/.768 Std. Error > 2 for Test #1 and 2.298 Kurtosis/.768 Std. Error > 2 for Test #3). This would suggest a homogenous distribution, making the detection of relationships harder. Yet, Test #1 and Test #3 were significant with the PLT. Test #1 was significant with the HMI. This suggests a very strong and precise relationship.

Comparison of procedural and object oriented performance
Comparing Test #1 (procedural concepts only) with Test #3 (object oriented concepts only) can be justified. The same students took both tests, were in the same class with the same instructor, and the same instructor graded both tests. The class environment was kept constant. The variances (influence) of intervening variables (sample, programming language, evaluation, teaching methods) were believed to be minimized. Treatments (instruction content) were different between Test #1 and Test #3. By using a step-wise regression analysis, a comparison of factors for Test #1, procedural content, and Test #3, OOP content, can be made.

Discussion Regarding the 2 independent variables: PLT and HMI
Since the data for this research were obtained after the course was over, there was the possibility that the Java course improved logical thinking skills. However, research has shown that a procedural programming course has no impact on logical thinking skills [34,40].

Based on these research studies, it was concluded that the Java course did not affect logical thinking skills, for Test #1 and part of Test #2. These two tests covered procedural concepts. Since cognitive development is fixed in adulthood (Kuhn et al., 1977) and not all adults develop to Formal Operations [4,23,47], it is suspected that the OOP of Java did nothing to enhance cognitive development as measured by the Propositional Logic Test (PLT).

Nevertheless, such research on the effects of learning OOP on cognitive development is warranted. OOP is a new set of programming languages, and found to be related to logical thinking skills as measured by the PLT. It has never been studied as improving logical thinking skills or formal operational cognition level.

Programming's relationship with cognitive development

The successful programmer must have an aptitude for logic. Logic is used in procedural programming [20,21]. Propositional logic has a direct analogy in computer programming [19]. Propositional logic concepts are found in procedural programming. The ability to do propositional logic is part of Piaget's formal operational stage [9,15]). The Propositional Logic Test (PLT) is an instrument that measures logic found in Piaget's Formal Operations and it correlates significantly with programming grades [40]. Formal operational reasoning ability is necessary for success in procedural computer programming/logic [2,11,18,26,27,31,44,46]. This study, using a university sample, has confirmed these prior findings of programming and logic with Test #1 and Test #2.

OOP is only recently being investigated. This study has shown with Test #3 ($r = .369$, $p < .027$) that Java, an OOP language, also requires Formal operational logic abilities. Yet, both Test #2 and Test #3 lacked IF/THEN/ELSE propositional statements. Concepts involving truth tables or propositional logic were never evaluated by these two OOP tests. Formal operational cognitive level deals with the ability to deal with abstractions. In this case, the PLT indicated one's ability to deal with abstractions either through logical programming abstract steps (Test #1) or with the manipulation of programming abstract objects (Test #3) or both (Test #2). Formal operational cognitive level is required to learn OOP or procedural programming. This is consistent with other research using procedural programming languages [19,26,31]. It is also consistent with research using C++ [52].

Programming's relationship with Hemispheric style

HMI results for Test #1, procedural concepts, from this research were consistent with prior research. Successful students in procedural programming use their left-brain [32,36,39]. HMI was not related to tests containing OOP concepts (Test #3, totally OOP content, and Test #2, partly OOP content). The Null Hypothesis #2 was not rejected for object oriented concepts. This is consistent with research using C++ [52].

Yet, Test #3 was related to the PLT, propositional logic found in the left side of the brain. This is what is interesting. It appears that OOP is independent of cognitive style since no significant correlation was found. A right-brain thinker does just as well as a left-brain or whole brain thinker with OOP.

Even though someone is right hemispheric dominant, the logical ability area of the left-brain may still be strong enough to do OOP since cognitive development is at formal operations. This may explain why right-hemispheric dominant students can be successful in OOP. Another reason why OOP may be friendly to both sides of the brain, is that object oriented properties are present in user cognition [28].

Summary

The HMI was selected in the regression analysis for Test #1, procedural concepts. It was never used with Test #3, object oriented concepts. OOP appears to be independent of hemispheric learning styles. There was no correlation between the HMI and PLT. The HMI may have a lesser relationship with different programming languages when compared with the PLT. It appears that logical aptitude, as measured by the PLT, is of greater importance in learning object oriented programming than cognitive hemispheric dominance as measured by the HMI. The relationship may be very small.

The findings of this study using Java are consistent with a study using C++ [52]. Both studies support the theory of the relationships between cognitive requirements of computer programming languages and programmers' cognitive characteristics [53].

Conclusions

The language paradigm shift from structured programming (procedural) to OOP thinking is difficult [17,34,48]. Procedural programmers find OOP difficult to learn. Yet OOP appears to be cognitive hemispheric friendly [28,52].

The problem with the shift from procedural to OOP appears not to be with any cognitive differences, developmental or hemispheric style. In fact, OOP may be easier since it is hemispheric friendly, uses less logic/truth table algorithms, and human cognition has object oriented properties.

Limitations of this study

There are limitations in any research that attempts to study the effects of independent variables on dependent ones in a controlled environment. The control of the environment is the first limitation we can identify with our study. The subjects were students who were free to attend class or not, get help in writing computer programs and prepare for the exams, each on their own. So there are potentially a variety of intervening and uncontrollable events that may affect a student's performance on an exam (the dependent variable).

We only collected and analyzed data for one semester. The semester-long course did not cover advanced OO topics in depth but it covered just enough to be able to distinguish the OO programming from procedural programming. Our results cannot be generalized much beyond our classroom environment. In defense of our study, however, that is how scientific theory building has to be done. Through repeated experimentation and variation of the values of the variables involved, a body of theory can be solidified.

The time laps between the exams during the semester, was short. The disadvantage of this is that a longer exposure to the treatment under a controlled environment (which is difficult to arrange) would give more statistical precision and validity to the results. Conversely, when the duration of the treatment is short, the effects of potential intervening variables that would be present in a longer treatment are avoided.

Recommendations for future research

An explanation for procedural programmers' difficulty with OOP may be due to proactive inhibition. This occurs when prior learning inhibits new learning [25]. The more similarity there is between the prior and the new learning, the greater the interference created by the old learning [37]. Bellin's (1992) research supports this conclusion by showing that students without any prior experience in structured programming techniques, performed better with object oriented concepts. Pennington et al. [41] found that procedural practices crept into objects (i.e. input, process, output design). In order to be more successful, procedural programmers may need to unlearn past programming practices such as action on data structures before learning OOP. Future research in this area is warranted.

Another area of research might be to determine whether it is better to teach object oriented concepts before procedural concepts. Bellin's (1992) object oriented concepts study showed that students' performance was better without any prior experience in structured techniques. Would the same be true if structured techniques were taught after object oriented concepts?

In this study, prior course background was considered, but no relationship was found. This may be due to almost all of the students having a high school programming course. Very few had a previous college programming course. There is the possibility that forgetting occurred over time. If the diversity of backgrounds were greater, different results might have been found. To determine whether prior courses can better prepare students for later programming courses or act as an inhibitor for a later programming course, will be difficult. It is infeasible to assign students randomly to semester programming courses. Students have a set curriculum of courses to follow. Also, prior courses may induce proactive interference or only indicate programming ability.

OOP concepts, a new programming language paradigm, were found to be related to logical thinking skills. Although related, OOP has yet to be shown to improve logical thinking skills or formal operational cognition level. Such research has been done with procedural programming. The findings for procedural programming indicate that logical thinking skills are not affected by learning procedural programming. As with procedural programming, research concerning the effects of learning OOP on logical thinking skills is warranted.

Future research in this area can continue to study the effects of learning Procedural and OOP, on the subjects' Hemispheric Style and Logical Reasoning. The fundamental research question is whether a person's hemispherical style can be affected by learning to program a computer using either a procedural or OO language. Likewise, the parallel research question is whether a person's Piagetian Logical Reasoning level can be affected (increased) by teaching that person to program a computer using a Procedural or OO language. In order to build and validate such a body of theory, a lot of controlled studies have to be performed to establish effects and relationships among the variables involved.

The long term implications of this line of research would be to build a body of theory that can help us advice people as to what fields of study they are inclined to be more successful in, based on the cognitive characteristics and aptitudes that each persona brings to bear.

References

[1] A.W. Andrew, The differential roles of right and left sides of the brain in memory formation, Behavioural Brain Research 98(2) (1999) 289–295.

[2] A. Azzedine, The relationship of cognitive development, cognitive style and experience to performance on selected computer programming tasks: An exploration, Dissertation Abstracts B48(6) (1987) 1799.

[3] R.J. Barker and E.A. Unger, A predictor for success in an introductory programming class based upon abstract reasoning development, *Proceedings of the 14th SIGCSE Technical Symposium on Computer Science Education of the ACM*, (Orlando, Florida, 1983).

[4] S. Bastian, J. Frees, L. Gruber, J. Johnson, B. Landes, L. Morton, S. Rozgony and J. Stewart, Are I.S.U. Freshman students operating at a formal level of thought processes? Contemporary Education 44 (1973) 358–362.

[5] D. Baum, The rough road to objects, Computerworld 27(31) (1993) 81.

[6] D. Baum, Objects go mainstream, Informationweek 590, (1996) 1A

[7] D. Bellin, A seminar course in object oriented programming, SIGCSE Bulletin 24(1) (1992). 134–137, Cited in R. Krovi and A. Chandra, User cognition representation: The case for an objet oriented model. Journal of Systems and Software 43(3) (1998) 165–176.

[8] R.F. Biehler and J. Snowmand, *Psychology Applied to Teaching*, 5th Ed., (Houghton Mifflin Company, Boston, Mass., 1986), p. 63.

[9] C.J. Brainard, *Piaget's Theory of Intelligence.* (Prentice-Hall, Englewood Cliffs, NJ, 1978).

[10] M.P. Bryden, Choosing sides: The left and right of the normal brain, Canadian Psychology 31(4) (1990) 297–309.

[11] R. Cafolla, The relationship of piagetian formal operations and other cognitive factors to computer programming ability (Development), Dissertations Abstracts A47(7) (1987) 2506.

[12] E. Chiapetta, A review of Piagetian studies relevant to science instruction at the secondary and college level, Science Education 60 (1976) 253–261.

[13] E.R. Doke and C. Hardgrave, *An Introduction to Object COBOL*, (John Wiley & Sons Publishers, 1998).

[14] T. Elrad, R.E. Filman and A. Bader, Aspect-oriented programming, Communications of the ACM 44(10) 2001, 28–32.

[15] M.A. Enyeart, Relationships among propositional logic, analytical reasoning, and Piagetian level, Dissertation Abstracts International A41(09) (1981) 397.

[16] H. Epstein, Stages in human mental growth, Journal of Educational Psychology 82 (1990) 876–880.

[17] J. Fedorowicz and A.O. Villeneuve, Surveying object technology usage and benefits: A test of conventional wisdom, Information Management 35(6) (1999) 331–334.

[18] S.H. Fletcher, *Cognitive Abilities and Computer Programming.* EDRS(ED259700), (1984).

[19] M.J. Folk, Influences of developmental level on a child's ability to learn concepts of computer programming, Dissertation Abstracts International 34(3) (1973) 1125a.

[20] A. Galton, Logic as a formal method, The Computer Journal 35(5) (1992) 431–440.

[21] H. Gibbs and A.B. Tucker, A model curriculum for a liberal arts degree in computer science, Communications of the ACM 29(3) (1986) 202–210.

[22] J. Greenbaum, Packaged applications, Software Magazine 17(7) (1997) 91–94.

[23] D.H. Griffiths, The study of the cognitive development of science students in introductory level courses, ERIC(ED096108) (1973).

[24] R. Guerraoui, Strategic directions in object-oriented programming, ACM Computing Surveys 28(4) (1996) 691–700.

[25] E.R. Hilgard and G.H. Bower, *Theories of Learning* (3rd ed). (Meredith Corporation, New York, 1966) 312–313.

[26] M.A. Hudak and D.E. Anderson, Formal operations and learning style predict success in statistics and computer science courses, Teaching of Psychology 17(4) (1990) 231–234.

[27] D.M. Irons, Predicting programming performance in novice programmers by measures of cognitive abilities), Dissertation Abstracts B43(4) (1982) 1283.

[28] R. Krovi and A. Chandra, User cognitive characteristics: The case for an object oriented model, Journal of Systems and Software 43(3) (1998) 165–176.

[29] D. Kuhn and J. Langer, L. Kohlberg and N. Haan, The development of formal operations in logical and moral judgment, Genetic Psychology Monographs 95(1) (1977) 97–188.

[30] M.G. Lieberman, *The Hemispheric Mode Indicator Technical Notes*, EXCEL, Inc, Barrington, Il, (1986).

[31] L.F. Little, The influence of structured programming, gender, cognitive development and engagement on the computer programming achievement and logical thinking skills of secondary students. Dissertation Abstracts A45(6) (1984) 1708.

[32] C.L. Losh, The relationship of student hemisphericity to performance in computer programming courses, Dissertation Abstracts A44(7) (1984) 2127.

[33] M.G. Mains, The effects of learning a programming language on logical thinking skills, Unpublished Thesis, University of Nevada, Las Vegas, Nevada, (1997).

[34] M.L. Manns and H.J. Nelson, An exploration of schema development in procedure oriented programmers learning object-oriented technology, in: *Proceedings of the fourteenth International Conference on Information Systems*, (1993) 385–386.

[35] B. McCarthy, *The Hemispheric Mode Indicator.* (Barrington, Il: Excel, Inc, 1986).

[36] M. Monfort, S.A. Martin and W. Frederickson, Information-processing differences and laterality of students from different colleges and disciplines, Perceptual and Motor Skills 70(1) (1990) 163–172.

[37] C.G. Morris, *Psychology, an Introduction*, Prentice-Hall, Inc. *Englewood Ciffs*, New Jersey, (1973), 170–172.

[38] M.G. Morris, C. Speier and J.A. Hoffer, An examination of procedural and object-oriented systems analysis methods: Does prior experience help or hinder performance? Decision Sciences 30(1) (1999) 107–136.

[39] C.F.P. Ott, Predicting achievement in computer science through selected academic, cognitive and demographic variables, Dissertation Abstracts A49(10) (1989) 2988.

[40] B. Owens and J. Seiler, A study of the relationship between performance in propositional logic and computer science, Proceedings of the Seventh Annual south Central Small College Computing Conference, The Journal of Computing in Small Colleges 11(7) (1996) 90–93.

[41] N. Pennington, A.Y. Lee and B. Rehder, Cognitive activities and levels of abstraction in procedural and object-oriented design, Human-Computer Interaction 10(2/3) (1995) 171–226.

[42] J. Piaget, Intellectual evolution from adolescence to adult, Human Development 15 (1972) 1–12.

[43] W. Price, Elements of Object-Oriented COBOL, 2nd Ed. (Object-Z Systems Publishers 2001).

[44] C.M. Ricardo, Identifying student entering characteristics desirable for a first course in computer programming, Dissertation Abstracts A44(1) (1983) 96.

[45] S. Riley, Is learning Cobol now a good long-term investment? Infoworld 20(44) (1998) 104.

[46] M.H. Schroeder, Piagetian, Mathematics and Spatial Reasoning as Predictors of success in Computer Programming, Dissertation Abstracts A39/08 (1979) 4850.

[47] M. Schwebel, Formal operations in first year college students, Journal of Psychology 91 (1975) 133–141.

[48] D. Shaffer, Predicting success in the undergraduage Introductory Computer Science course using the Theory of Planned Behavior. Unpublished Dissertation, University of Texas, Ausin, (1990).

[49] S. Sheetz and G. Irwin, D. Tegarden, H.J. Nelson, and D.E. Monarchhi, Exploring the difficulties of learning object-oriented technique, Journal of Management Information Systems 14(2) (1997) 103–131.

[50] T. Supprian and E. Hofmann, The fornix of the human brain: Evidence of left/right asymmetry on axial MRI scans, Surgical and Radiologic Anatomy 19(2) (1997) 105.

[51] A. Topper, *Object-Oriented Development in COBOL.* (McGraw Hill Publishers, 1995).

[52] G. White, Cognitive characteristics for learning C++, Journal of Computer Information Systems 42(3) (2002) 51–55.

[53] G. White and M. Sivitanides, A theory of the relationships between cognitive requirements of computer programming languages and programmers' cognitive characteristics, journal of information systems education 13(1) (2002) 59–66.