Effects of a Case-Based Reasoning System on Student Performance in a Java Programming Course

Cecil Schmidt
Computer Information Sciences
Washburn University
Topeka, Kansas 66621, USA
cecil.schmidt@washburn.edu

ABSTRACT

The purpose of this study was to determine if a case-based reasoning tool would improve a student's understanding of the complex concepts in a Java programming course. Subjects for the study were randomly assigned from two sections of an introductory Java programming course. Posttests were used to measure the effects of the case-based reasoning tool (CBJava) on learner competency. Results of the study using a Mann-Whitney *U* test indicated a significant difference between the group who used CBJava on complex questions and the group who did not ($m\ rank = 11.50$; $U = 3.500$, $p < .05$, $M = 22.71$ vs. $M = 17.88$). No significant difference was indicated between the groups on simple questions. Recommendations from this study include supporting complex content through examples, providing a case-based instructional aid for complex topics, and extending CBJava's framework to support other courses and disciplines.

Keywords: Distance Education, Java, Case-based Reasoning

1. INTRODUCTION

1.1 Background

Java, an object-oriented programming language typically used as the language of choice in a first programming class (CS1), is complex and difficult for students to learn. Raab, Rasala, and Proulx (2000) suggest that the cross-platform capabilities of Java and the robust graphical user interface (GUI) components provide a great argument for using Java to teach programming, however the complexity of building a complicated GUI in a CS1 course is problematic. Raab et al. suggest using a toolkit of pre-developed classes that can be used as the framework for beginners to build from to reduce this complexity. Similarly Kolling (1999) reported that educators found the lack of an adequate Integrated Development Environment (IDE) that could be used as a learning aid was a major problem in teaching Java. BlueJ was developed to address these needs (Kolling, Quig, and Patterson, 2003; Kolling 2004). Another major difficulty with teaching Java is that difficult concepts must be addressed at an early stage (Biddle and Tempero, 1998). Even the writing of a simple one-line program in Java requires the introduction of complex concepts such as inheritance, static methods, or exceptions. The complexity (breadth of the language) and instability (changes and additions to the language) of teaching an introductory computer science programming course is also documented by Roberts (2004). The essential complexities of Java include encapsulation, inheritance, polymorphism, reuse, etc., whereas the unnecessary complexities include the magnitude of the Java 2 class libraries (some 50,000 library functions) and the rapid obsolescence of libraries and tools that are available for Java. Unnecessary complexities are also illustrated by the differences in the size of the textbooks that are now used to teach Java. One of the more popular books (Deitel and Deitel, 2003) has 1536 pages of text whereas the classic *Pascal User Manual and Report* (Jensen and Wirth, 1991) that was used to teach Pascal had about 226 pages.

1.2 Scaffolding Student Understanding with CBR

Case-based reasoning (CBR) is a learning model (Schank, 1982; Kolodner, 1993) and problem solving paradigm (Aamodt and Plaza, 1994; Leake, 1996; Mitchell, 1997) that incorporates problem solving, understanding, and learning and integrating them with memory processes. CBR is a constructivist learning theory which suggests knowledge building (Bruner, 1996) from our previous experiences through access to prior cases for both reuse and adaptation. Both new and adapted cases can be stored for future use, thus the learning occurs as a natural consequence of reasoning. Finally, CBR, in the context of a learning theory, is also tightly integrated as part of another constructivist learning theory referred to as the Cognitive Flexibility Theory (Spiro, Coulson, Feltovich, & Anderson, 1988; Spiro, Feltovich, Jacobson, & Coulson, 1992; Spiro and Jacobson, 1995). The Cognitive Flexibility Theory (CFT) suggests that advanced learning in ill-structured [complex] domains must be supported through alternative cases and multiple, crisscrossing paths through a set of knowledge content. CFT suggests that the complexity of these types of domains cannot simply be understood in a single pass.

Providing the student with a tool for learning Java that is guided by the CBR constructivist learning model should provide the scaffolding to support the learning of the difficult concepts encountered with the Java programming language. Examples can be used to demonstrate cohesion between the concepts and features of the language and real-life problems. Students will learn by example as well as adapt new examples. One approach to providing this type of system is through a web-based hypertext learning environment where the leaner is provided with concrete examples, i.e. actual cases, rather than abstract rules, explicitly integrating memory, learning, and reasoning, i.e. a CBR-grounded learning environment (Kolodner and Guzdial, 2000). Experience is provided by means of a case library that has cause, effect, and lessons learned components. Learners may access these cases through multiple indexes that crisscross the content as prescribed by the CFT. Incorporating an appropriate set of indexes over these experiences provides the learner with alternative views into the same sets of cases. Content should be organized in way that a "learner sees a range of conceptual applications close together, so that conceptual variability can be easily examined" (Spiro et al., 1992, p. 68).

1.3 Purpose

I now report on a study of the effects of a case-based reasoning system as a support for learning the complex concepts in an object-oriented programming course. The primary purpose of this study was to determine if a CBR-grounded tool will improve a student's understanding of the more difficult concepts in an object-oriented programming course. At the time of this study there was no public record of this particular research.

Below is a list of the research questions that were addressed by this study. The categorization of simple and complex questions is addressed in the methods section of this paper.

1. Is there a statistically significant difference in the performance on simple questions between the Case-Based Reasoning Assisted (CBA) group and the Lecture Notes Only (LNO) group?
2. Is there a statistically significant difference in the performance on complex questions between the CBA group and the LNO group?

2. METHOD

2.1 Participants

Two sections of the Object-Oriented Programming 1 (OOP-1) course in Spring 2005 were used to represent the sample. Only two sections were used because there were only two sections of OOP-1 offered at the study site. OOP-1 was an introductory course in Java which emphasized object-oriented programming and design. It was the second required programming course in both the Bachelor of Science degree in Computer Information Systems and the Bachelor of Arts degree in Computer Information Science at an NCAA, Division II university located in the Midwest region of the United States. Division II universities tend to be smaller public or private universities and this particular university has an approximate average enrollment of 7800 students. Students typically take this course during the second semester of their freshman year.

Of the two OOP-1 sections, one was face-to-face and the other was an online distance-education course. The face-to-face section normally met two days per week in a lecture setting. The online section was handled in an asynchronous manner where the students attended the class virtually by accessing the course content through WebCT (www.webct.com). Students in both sections had the same deadlines for both programming labs and tests. During the period of the research all students were attending the course as if they were in the online section. The experiment began in the seventh week of class and ran for three weeks.

Twenty-one students were initially enrolled in the two sections of Object-Oriented Programming 1. Of these, 11 students were enrolled in the online section and 10 students were enrolled in the face-to-face section. Prior to the beginning of the study, five students dropped the course. Of the remaining 16 students, all signed the consent form agreeing to participate in the study. These 16 students were randomly assigned to one of two treatment groups: Group 1 or Group 2. Before the completion of the experiment one student from Group 1 was dropped from the study because of the student failed to take both posttests.

All students received credit for using the CBJava tool (see section 2.2.2 for description of CBJava) as an incentive however this score was not factored into the study. To receive credit the student was required to create a content area example and post it to CBJava. All postings were anonymous to the other students but could be traced backed to the student by the researcher. Before receiving the CBA (CBJava assisted) treatment, students were provided with a training area within CBJava that was not part of the study. The training area set up for the students was composed of the decisions content area. Students were required to sign on to the CBJava site and post an example to the training area. This training occurred two weeks prior to the actual study. All students took part in this training.

2.2 Materials

2.2.1 Posttests: Two posttests, Posttest 1 and Posttest 2, were given immediately following the treatment conditions. Each test contained a set of questions covering the content areas addressed during the respective period in the study. These questions were categorized according to Bloom's taxonomy of learning objectives (Bloom and Krathwohl, 1956). Two categories of questions were created, simple questions and complex questions. Simple questions were those questions which measured the learning objectives of knowledge, comprehension, and application. The complex questions were those questions which measured the learning objectives of analysis, synthesis, and evaluation. A set of candidate questions for each of the two tests was generated by the researcher who pulled candidate questions from the normal assessment tests given in previous semesters. These candidate questions were then provided to two other faculty members who had previous experience in teaching a Java programming course. Each of these faculty members as well as the researcher classified the questions as either simple or complex. Questions were then classified by majority vote. Additionally, questions which were unclear were either clarified or dropped as candidate questions. A breakdown of the questions by format and their categorization as either simple or complex is provided in Table 1.

Question Format	Number	Category	Points
True/False	11	Simple	11
True/False	1	Complex	1
Multiple Choice	7	Simple	14
Short Answer	6	Complex	24

Table 1. Categorization of Questions by Format

Both posttests had an identical format and ordering of questions. A sub score for each question type (category) was generated based on the total points scored on the related assessment questions.

Finally, although the short answer questions had a greater point value per question, partial credit was given for partially correct answers. In order to ensure consistency in grading, a list of reasons for the partial credit along with the amount of partial credit awarded was maintained and used as a guide for awarding points.

2.2.2 CBJava: CBJava is a CBR-grounded hypertext system that was developed to be used as an instructional content aid for students who are learning Java (Schmidt, 2004, 2006). The design of this system is similar to the hyperbook design used in the *Engines for Education* web site authored by Schank and Cleary (1995). In particular the foundational architecture of CBJava was its question and answer interface implemented in hypertext that was available on the World Wide Web. In addition to being a hyperbook, this site provided students the ability to add their own examples. As these examples were added, an expert (in this case the researcher) rated the examples based upon quality and context. In this way a case-base of validated examples were made available to other students for further learning and research.

CBJava's primary knowledge acquisition process involved the submission of new Java examples by the student and expert review and validation performed by the instructor (the researcher). Indexing of the example was performed by the student through a Web interface. During the study only one of the content areas within CBJava was open at a time. Thus the indexing of the example was limited to that particular content area. For example those students who were given the CBA treatment during the first period of the study could only index their examples under object design. Those students had no access to the inheritance content area. During the second period of the study those students who were given the CBA treatment could only index their examples under inheritance. Again, those students had no access to the object design content area.

At the time the example was submitted the example had a note stating that it has not been validated. On a daily basis the instructor reviewed the submissions and either accepted the submission or revised it. In the revision cycle, the instructor identified the improvements and classified the original example. Both the before (as submitted) and the after (post review) versions were retained.

2.3 Procedure

2.3.1 Research Design: This study involved characterizing the sample based on learner competency assessment questions categorized according to Bloom's taxonomy of learning objectives (Bloom and Krathwohl, 1956). Independent variable *A* was defined as instructional support (case-based hypertext learning tool versus lecture notes only) and independent variable *B* was defined as the question type (simple assessment questions that measure the lower levels of learner competency and complex assessment questions that measure the higher levels of learner competency). The experimental design is shown in Figure 1. In the experimental design depicted in Figure 1 Group 1 and Group 2 refer to the groups of students randomly selected from two sections of Object-Oriented Programming 1. Object Design and Inheritance are the two content areas that had instructional support. The treatments X(CBA) and X(LNO) refer to the case-based hypertext learning tool support and the lecture notes only support. Posttest 1 and Posttest 2 refer to the two posttests that were given. The instruments used to collect the data for analysis were known up front and, for the most part, had been validated in prior research.

	Object Design	Posttest 1	Inheritance	Posttest 2
Group 1	X(CBA)	O	X(LNO)	O
Group 2	X(LNO)	O	X(CBA)	O

Figure 1. Experimental Design depicting Groups, Treatments, and Observations

2.3.2 Treatment: Lecture content to members of both treatment groups 1 and 2 was provided in the form of hypertext videos that were administered through WebCT. The recordings were developed using sofTV (www.sofTV.net). Each of these recordings was placed into WebCT and integrated through a hypertext document. Both groups also shared an online space in WebCT. All lecture-notes, online discussions, and homework assignments were also provided and administered through WebCT to both groups. Email was handled externally using the study sites' email system.

Two complex and ill-structured content areas had additional instructional support through a case-based reasoning tool called *CBJava* (Schmidt, 2004). These content areas were object design and inheritance. During the coverage of object design, Group 1 was required to use the CBJava tool, that is, the CBA treatment. Group 2 received no assistance from CBJava during this period, that is, the LNO (lecture notes only) treatment. After completing the coverage of object design, Posttest 1 was given to both groups. The duration for this part of the experiment was one and one half weeks, culminating with the Posttest 1.

Question Type × Content Area	Group 1 Mean	N	SD	Group 2 Mean	N	SD	Total Mean	N	SD
Simple									
Obj. Des.	22.00	7	2.449	19.63	8	4.340	20.73	15	3.674
Inheritance	20.71	7	2.289	18.00	8	2.976	19.27	15	2.939
Complex									
Obj. Des.	22.71	7	1.890	17.88	8	3.563	20.13	15	3.758
Inheritance	20.57	7	3.505	15.63	8	3.852	17.93	15	4.383
Total									
Obj. Des.	44.71	7	3.450	37.50	8	5.682	40.87	15	5.927
Inheritance	41.29	7	5.282	33.63	8	5.476	37.20	15	6.527

Table 2. Descriptive Statistics for Test Scores for Question Type by Content Area by Group

Inheritance was covered immediately following the unit on object design. During the coverage of inheritance, Group 2 was required to use the CBJava tool, that is, the CBA treatment. Group 1 received no assistance from CBJava during that period, that is, the LNO treatment. After completing the coverage of inheritance, Posttest 2 was given to both groups. The duration for this part of the experiment was one and one half weeks culminating with the Posttest 2.

Both posttests were administered on the study sites' campus. Additionally these tests were proctored by a faculty member at the study site who was not the researcher. This faculty member coded both the tests and the surveys in order to protect anonymity during the study. Both posttests were scored by the researcher before they were matched back to the student in order to minimize bias.

2.4 Limitations of the Study

It was the goal of this study to do a quantitative analysis with as much statistical rigor as possible. The subjects used for this study were randomly assigned to one of two groups. However, the pool of subjects was limited to those students who had enrolled in Object-Oriented Programming 1 at an NCAA, Division II university located in the Midwest region of the United States. Therefore, it is more difficult to generalize this study to a much broader population, and that limits its external validity.

The ordering of the two types of instructional support is also a limitation of the study. The case-based hypertext tool was not used as an instructional aid until midway through the course. It could be argued that by that time the students do not require any additional support. They may have learned how to use the existing resources to support their learning. Thus, there may be no significant difference between the performance of the students with or without the case-based hypertext tool as an instructional aid. However, it is only at about the midpoint of the course where the concepts become more complex and ill-structured. So, introducing the case-based hypertext tool at that time was appropriate. Other sequencing situations arise as well, but because of the number of groups, the limitations of the sample sizes, and the ethical requirement to provide all students the same aids, this was the best that could be done.

Other extraneous variables may have affected the outcome of the study. In particular it was difficult to determine how much of the content within the case-based hypertext tool was actually read by each student. The only guarantee that a student accessed the tool was that they submitted the example. However, there was incentive for them to read the content in that it aided them in creating an example (which was required), and it helped them prepare for the graded posttest, therefore the impact to the results of the study were minimal. Finally, not all of the students submitted examples in a timely basis. In order to ensure that all of the subjects submitted an example, several directed emails were sent. No special coaching on creating the example occurred, therefore impact to the results of the study were also minimal.

Some of the limitations to the external validity were eliminated by limiting the differences in the treatment groups to one particular variable which was the type of instructional support. For the duration of the study the transmission of the course to all subjects was the same, that is, the transmission was online. One can argue that the viewing of a digital video of the lecture can be done anytime and as such is another variable in the experiment. However, for the purposes of this study the time and space dependencies were subsumed in the instructional mode.

3. RESULTS

3.1 Effects of CBJava

In order to answer the research questions, two types of non-parametric tests were performed on the sample as prescribed in the experimental design. A third test, a parametric test, was performed on the sample in order to better understand the results of the first two tests.

Descriptive statistics including means and standard deviations for each content area (Object Design and Inheritance) separated by question type (Simple and Complex) for each group (Group 1 and Group 2) are provided in Table 2. Notice that the mean scores on the posttests (Posttest 1 covered Object Design and Posttest 2 covered Inheritance) for Group 1 are consistently higher than those in Group 2. A higher mean score represents a better

Question Type × Treatment	Group 1			Group 2			Total		
	Mean	*N*	*SD*	Mean	*N*	SD	Mean	*N*	*SD*
Simple									
CBA	22.00	7	2.449	18.00	8	2.976	19.87	15	3.357
LNO	20.71	7	2.289	19.63	8	4.340	20.13	15	3.461
Complex									
CBA	22.71	7	1.890	15.63	8	3.852	18.93	15	4.728
LNO	20.57	7	3.505	17.88	8	3.563	19.13	15	3.681
Total									
CBA	44.714	7	3.450	33.625	8	5.476	38.800	15	7.272
LNO	41.286	7	5.283	37.500	8	5.682	39.267	15	5.650

Table 3. Descriptive Statistics for Test Scores for Question Type by Treatment by Group

performance on the posttest. The mean scores are the average number of total points for the group out of a maximum of 25 points for each part of the posttest (i.e., each posttest contained complex questions and simple questions each worth 25 points). Neither group did as well on the posttest covering inheritance (Posttest 2) as they did on Posttest1.

Descriptive statistics including means and standard deviations for each level of instructional support treatment (CBA and LNO) separated by question type (Simple and Complex) for each group (Group 1 and Group 2) are provided in Table 3. Notice that the mean scores for Group 1 were consistently higher than Group 2 regardless of instructional support treatment.

The first non-parametric test performed was a Wilcoxon Signed Ranks test at each level of instructional support (CBA and LNO) at $p < .05$ level of significance. This test was performed for each question type (Simple and Complex). Matched pairs were created by matching the test scores of the same student across the two content areas (Object Design and Inheritance). The change in score was calculated by subtracting the student's score obtained with CBA instructional support from the student's score with LNO instructional support. A negative change meant that the student performed better with CBA support versus LNO support.

The results indicate no significant differences in the student test scores for either complex questions ($Z = 0.000, p > .05$), simple questions ($Z = -0.106, p > .05$), or the total set of questions ($Z = -0.378, p > .05$). Specifically the Wilcoxon test result ($Z = 0.000$) for Complex LNO – Complex CBA indicates that on the complex questions there were just as many students who performed better with lecture notes only (i.e., the LNO treatment) as those who performed better with the CBJava as an instructional support tool (i.e., the CBA treatment). This finding conflicts with the expectation that the CBA treatment should enable a student to perform better on complex questions than with the LNO treatment. One possibility for this finding was that that there may be a carryover effect for those students who received the CBA treatment in period 1 (during the coverage of Object Design) of the study. That is, the CBA treatment may have helped them sufficiently that they performed better than expected on the second assignment for which they did not have the tool available.

The carryover effect question (i.e., was there a carryover effect of the treatment) was answered once for each set of questions (i.e., Complex and Simple). The 2 × 2 mixed-design ANOVA was used because there is not an alternative non-parametric design that can test for carryover effects. In the mixed design ANOVA used in this study, the repeated measures variable was treatment (i.e., the ordering of treatments), and the independent variable was posttests.

The effects of the group are essentially the same as a sequence effect because of the ordering of the treatment. Recall that Group 1 was the group of subjects who received the CBA treatment in period 1 (Object Design) and the LNO treatment in period 2 (Inheritance). Group 2 was the group of subjects who received the CBA treatment in period 2 (Inheritance). There was no significant effect indicated by the posttest scores by students on complex questions for the Group × Treatment interaction ($F(1,13) = 4.452, p > .05$). The main effect for treatment on students' posttest scores on complex questions was also not significant ($F(1,13) = 0.003, p > .05$). The main effect for treatment finding is a similar result to the findings of the Wilcoxon Signed Ranks test (i.e., both the non-parametric test and the parametric test came to similar conclusion that there were no treatment effects). However there was a significant effect (i.e., a carryover effect) of treatment on Group 1 for complex questions ($F(1,13) = 12.718, p < .01$). These results indicate that there was a significant carryover effect of Treatment on students as indicated by test scores on complex questions from posttest 1 to posttest 2 for those students who received the CBA treatment in the first period. Students in Group 1 had a similar drop in mean scores (2.14 points) between posttest 1 and posttest 2 on complex questions as did Group 2 (2.25 points) even though it was Group 2 (not Group 1) who received the CBA treatment for the content area (object design) covered by posttest 2 (see Table 2).

There was no significant effect indicated by the posttest scores by students in Group 1 on simple questions for the Group × Treatment interaction ($F(1,13) = 3.28, p > .05$). The main effect for Treatment on simple questions was also not significant ($F(1,13) = 0.045, p > .05$). The main effect for this treatment finding as indicated by posttest scores on simple questions is a similar result to the findings of the Wilcoxon Signed Ranks test (i.e., both the non-parametric test and the parametric test came to similar conclusion that there were no treatment effects on students as indicated by posttest scores on simple questions). Finally there was no significant effect on simple questions for Group ($F(1,13) = 3.16$, $p > .05$). These results indicate that there was no significant carryover effect on simple questions for those students who received the CBA treatment in the first period.

The third test performed was a Mann-Whitney *U* test. A Mann-Whitney *U* test was used to compare two independent samples (Group 1 and Group 2) because of the small sample size (Huck, 2004, p. 496). Because the results of the 2 × 2 mixed-design ANOVA test indicated that there was a significant carryover effect of Treatment on students as indicated by test scores on complex questions, this third test was performed for period one only (during which the Object Design content area was covered). That is, the carryover effects confound the Mann-Whitney *U* test for period two (during which the Inheritance content area was covered), therefore it was not performed for period two.

The Mann-Whitney *U* test was calculated examining the treatment effects (CBA versus LNO) on students as indicated by test scores on the various types of test question (Complex and Simple) in period one (Object Design). Primarily this test was utilized to further test the treatment effects on complex questions because this is where the carryover effect occurred; however, for completeness, both simple and the combined set (Simple and Complex) of questions were tested as well.

Students in Group 1 (those who received the CBA treatment first) performed significantly better on the complex questions (m rank = 11.50; $U = 3.500$, $p < .05$, $M = 22.71$ vs. $M = 17.88$) than those students in Group 2 (those who received the LNO treatment first). These same students (Group 1) did no better than the students in Group 2 on simple questions (m rank = 9.21; $U = 19.500$, $p > .05$, $M = 22.00$ vs. $M = 19.63$). However the students in Group 1 did perform significantly better on the total set of test questions (m rank = 11.29; $U = 5.000$, $p < .01$, $M = 44.71$ vs. $M = 37.50$). The results of the Mann-Whitney U test indicate that the students who were provided with the CBJava tool in the first period of the study (i.e., the CBA treatment) performed significantly better than those students who were only provided with the lecture notes (i.e., the LNO treatment).

3.2 Student Background Assessment Results

A one-tailed Spearman *rho* correlation coefficient was calculated for the relationship between subjects' grade in background coursework (CM111 and PH110) and their test scores for each of the content areas (Object Design and Inheritance) broken out by question type (Simple and Complex). A Spearman *rho* correlation was used because grade is an ordinal measurement. Descriptive statistics for background coursework including means and standard deviations for each group (Group 1 and Group 2) as well as the combined groups are provided in Table 4. The mean scores reported are the average letter grades of the students based on a scale of A = 4, B = 3, C = 2, D = 1, and F = 0.

Although the mean scores for students in Group 1 appear to be better than Group 2, it is only by chance that this occurred. The differences in mean scores does raise a question about the significant effect of the CBA treatment on Group 1 as indicated by scores on posttest 1 found by the Mann-Whitney *U* test. However, the Mann-Whitney *U* test is a ranked-based test, not a means-based test (Huck, 2004, p. 496). It is also a distribution free test (i.e., a normalized distribution is not assumed). Thus, the question about the results of the Mann-Whitney *U* test (i.e., the significant effect of the CBA treatment on Group 1) is mitigated. Still, the differences in student background should be considered when analyzing the total set of results of this study.

Relevant bivariate correlations for Group 1, Group 2, and the combined groups are provided in Table 5. Of particular interest was the strong positive correlation found between the background knowledge and the total test scores for both content areas for the combined groups. This finding suggests that students who did well on their background coursework scored well on the posttests, and students who did poorly on their background coursework scored poorly on the posttests. This finding, once again, also raises the question about the significant effect of the CBA treatment on Group 1 as indicated by scores on posttest 1 found by the Mann-Whitney *U* test. Since Group 1 students performed better in their background coursework and because there was a strong positive correlation found between background knowledge and the total test scores, it could be argued that Group 1 students would have most likely performed better on the posttest 1 than Group 2 even without the CBA treatment. Again, this argument can be answered. Notice that for the first period (the period tested by the Mann-Whitney *U* test) there was no significant correlation found for Group 1 between Complex questions and either CM111 ($p = .107$) or PH110 ($p = -.095$). This was also the case for Group 2 (i.e., for CM111, $p = -.057$ and from PH110, $p = .463$). Recall that the Mann-Whitney *U* test found that students in Group 1 (who received the CBA treatment) performed significantly better than students in Group 2 (who received the LNO treatment) on complex questions but there were no differences found on the simple questions. Thus, even if one argues that Group 1 appears to have had better background knowledge than Group 2, this difference does not appear to have influenced the findings of the Mann-Whitney *U* test.

4. DISCUSSION

4.1 Effects of CBJava on Simple Questions

The first research question dealt with student performance on simple questions with and without the use of CBJava. The results of three separate tests, a Wilcoxon Signed Ranks test, a 2 × 2 mixed-design ANOVA test, and a Mann-Whitney U test, found that there was no significant difference between the two groups on simple questions. These findings are consistent with the research by Kozma (1994). His research suggests that effective use of technology is one that is grounded in the "cognitive and social processes by which knowledge is constructed". The simple questions that were tested in this study were ones that fell on the lower end of Bloom's taxonomy of learning objectives (Bloom and Krathwohl, 1956). Simple questions are typically ones that

	Group 1			Group 2			Total		
Course	Mean	*N*	*SD*	Mean	*N*	*SD*	Mean	*N*	*SD*
CM111	3.67	6	.516	3.13	8	.835	3.36	14	.745
PH110	2.86	7	1.464	2.38	8	1.188	2.60	15	1.298

Notes. One subject did not take CM111 in Group 1.

Table 4. Descriptive Statistics of Background Coursework by Group

		Object Design			Inheritance		
Group	Course	Simple	Complex	Total	Simple	Complex	Total
Group 1	CM111	.853*	.107	.735*	.414	.335	.414
	PH110	.667	-.095	.406	.378	.735*	.655
Group 2	CM111	.139	-.057	.255	.513	-.026	.207
	PH110	.199	.463	.281	.168	.444	.279
Combined	CM111	.497*	.335	.542*	.549*	.330	.519*
	PH110	.480*	.429	.477*	.461*	.592*	.561*

*$p < .05$

Table 5. Bivariate Correlations between Background Courses and the Posttest Scores for each Content Area for Group

do not require a lot of deep thought. Quite often the content covered in a face-to-face lecture does not even address these types of questions. Most instructors would make the assumption that a college student should be able to gain the necessary understanding from assigned readings and homework with minimal teacher-learner interaction in order to answer simple questions. Thus, providing a learning aid such as CBJava that is designed to support advanced learning in complex and ill-structured domains would not be expected to provide significant benefit in this area. The findings of this study were consistent with the existing research such as that of Spiro et al. (1992).

4.2 Effects of CBJava on Complex Questions

The second research question dealt with student performance on complex questions with and without the use of CBJava. The first test conducted to address this question, the Wilcoxon Signed Ranks test, found no significant difference on complex questions between the two groups. This finding might reasonably be explained by the students' reaction to carryover effects. The possibility of carryover effects was considered because access to the CBJava tool in the first period might have provided the subject with an improved understanding of the content of the subject on the first topic, and thus an a better understanding may have affected or "carried over" this knowledge into the second period. Thus, a second test, a 2 × 2 mixed-design ANOVA test for carryover effects on complex questions for the CBA treatment was performed.

The 2 × 2 mixed-design ANOVA test for carryover effects did indeed find a significant carryover effect on complex questions for those students who used the CBJava tool in the first period of the study. Thus, a third test was necessary in order to determine why the first test, the Wilcoxon Signed Ranks test, failed to find a significant difference. This third test was conducted on only the first period of the study because the second period was corrupted by the carryover effects. The results of the Mann-Whitney *U* test that compared the CBA group to the LNO group did find a significant difference on complex questions between the two groups on the first period. Based upon the findings of these three tests, the null hypothesis that there would be no significant difference between the two groups on complex questions was rejected.

The results of the Mann-Whitney *U* test showed that the students who, in the first period of the study, were provided with the CBJava tool to support their learning, performed significantly better than those students who were provided with only the lecture notes. The results also showed when considering the entire set of questions (Simple and Complex) the students who used the CBJava tool performed significantly better than those who only had access to the lecture notes, although they did no better on the simple questions. Interpreting these results a bit further, they show that the improvement on the complex questions was of such significance that the improvement drove the overall performance on the entire set of questions.

The complex questions that were tested in this study were questions that fell on the higher end of Bloom's taxonomy of learning objectives (Bloom and Krathwohl, 1956). Complex questions are typically ones that require sustained deep thought. Quite often the content covered in a face-to-face lecture primarily addresses these types of questions either by providing an alternative view of the content or through teacher-learner interaction. Typically complex content does require a significant amount of teacher-learner interaction.

The findings of this study support the alternative hypothesis that providing CBJava as an instructional support tool will improve student performance on complex questions. The findings are also consistent with the Cognitive Flexibility Theory (CFT) described by Spiro et al.

(1992). The basic premise of this theory and the related research is that complex subject matter can be learned best if it is provided with multiple views or indexes. CBJava provides this capability.

4.3 Conclusions

This study found that CBJava is an instructional tool that can significantly improve students' understanding of complex subject matter. Through its theoretical grounding in case-based reasoning and Cognitive Flexibility Theory, CBJava provides the additional scaffolding required by students to learn complex and ill-structured concepts. CBJava provides a question and answer interface that allows the student to learn through exploring the content. It avoids the rigidity found in textbooks through its hypertext implementation. And, as a case-based reasoning tool, CBJava enhances our ability to teach introductory programming classes whether or not they are online.

Providing a mechanism that allows students to submit examples, receive feedback, and see the examples of others supports learning through its constructivist grounding. This study found that students do make significant gains in their understanding of the subject matter when given this type of instruction. These finings are consistent with the learning theory of constructivism and its applications through CFT (Spiro and Jacobson, 1995).

Although expert feedback helps the student, there are costs involved that need to be considered. Expert feedback requires an investment of an instructor's time and talents. In a classroom that contains a large number of students, it may not be feasible to review examples from every student. However, these economic issues can be overcome.

By providing expert feedback on one example and making the original example, its repaired version, and the feedback available to the entire class, all students benefit. In a large classroom, a random set of submitted examples can be reviewed and repaired rather than the entire set. This would alleviate some of time burden on the instructor. Additionally the classroom size could be limited. In fact at the study site, the online class size limit is typically 24 students. However, limiting the class size also has economic ramifications. Alternatively a larger class size could be supported with the help of either teaching assistants, former students, or other experts that would be willing to perform the expert review and feedback.

It is reasonable to require online instructors to provide feedback to students. Traditionally this feedback comes in the form of emails, discussion threads, graded assignments, and graded tests. Providing expert review of posted examples is another alternative to this set of instructor-learner interactions from which the instructor can choose from in order to support student learning.

CBJava can be generalized to many different content areas, and it is not simply limited to object-oriented programming. Its underlying relational architecture is easily extendable to other domains with minimum technical support from a programmer. CBJava provides examples with multiple links to different components of the same example as well as links to multiple examples that illustrate the same concept. Some learners may find this difficult to use and understand, and this was found to be true in this study. However, this study found that although these students may have perceived this to be a distraction, in fact this "distraction" improved their learning.

4.4 Recommendations

Supporting complex content through examples that illustrate both good and bad alternatives is an important method of scaffolding students' understanding. Learning complex topics can be supported through examples where the learner can construct their own knowledge. Complex topics covered in an object-oriented programming course should be illustrated with both correct and incorrect examples. Both types of examples should have expert commentary tied to them with reasons why they are incorrect or correct.

Complex content requires enhanced instructor-learner interactions. These types of interactions are very difficult to replicate in an online course. Providing instructional support for complex content is important regardless of whether or not the course is online or face-to-face. However it is most problematic in the online course. An instructional support tool such as CBJava is therefore a necessary component of an online course. Alternatives such as threaded discussions may work as well, but do not allow for the natural learning that case-based tool such as CBJava provides.

Finally, economic costs to setup a course supported with a case-based tool based upon CBJava's framework need to be considered. Currently CBJava's framework is extendable to other courses, but its extension requires technical support from a programmer. Development of the web pages with multiple indexing schemes similar to those in CBJava does require a significant amount of the instructors' time and talents. Therefore, it might be more feasible to employ an instructional designer to assist with the initial setup of the tool rather than expecting the instructor to develop it themselves.

5. ACKNOWLEDGEMENTS

I would especially like to thank Dr. Diane McGrath of Kansas State University and Dr. Donna LaLonde of Washburn University for their inspiration and support of this research. Additionally I would like to thank Dr. Mike Mosier of the Mathematics and Statistics department at Washburn University for his review of the statistical analysis. No research effort can such as this can be accomplished without the help of others.

6. REFERENCES

Aamodt, A., & Plaza, E. (1994), "Case-based Reasoning: Foundational Issues, Methodological Variations, and System Approaches." AI Communications, 7(1), 39-59.

Biddle, R., & Tempero, E. (1998), "Java Pitfalls for Beginners." SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education), *30*(2), 48-52.

Bloom, B. S., & Krathwohl, D. R. (1956), Taxonomy of Educational Objectives; The Classification of Educational Goals, by a Committee of College and University Examiners (1st ed.), New York: Longmans, Green.

Bruner, J. (1996), The Culture of Education, Cambridge, MA: Harvard University Press.

Deitel, H. M., & Deitel, P. J. (2003), Java How to Program (5th ed.), Upper Saddle River, NJ: Prentice Hall.

Huck, S. W. (2004), Reading Statistics and Research (4th ed.), Boston: Pearson.

Jensen, K., & Wirth, N. (1991), Pascal User Manual and Report (3rd ed.), New York: Springer Verlag.

Kolling, M. (2004), "BlueJ – the Interactive Java Environment." Retrieved June 30, 2004 from http://www.bluej.org

Kolling, M. (1999), "The Problem of Teaching Object-Oriented Programming Part II: Environments." Journal of Object-Oriented Programming, 11(9), 6-12.

Kolling, M., Quig, B., & Patterson, A. (2003). "The BlueJ System and its Pedagogy." Computer Science Education, 13(4), 249-268.

Kolodner, J. (1993), Case-based Reasoning. San Francisco, CA: Morgan Kaufmann.

Kolodner, J. L., & Guzdial, M. (2000), "Theory and Practice of Case-based Learning Aids." In D. H. Jonassen, & S. M. Land (Eds.), Theoretical Foundations of Learning Environment, (pp. 215-242). Mahwah, NJ: Lawrence Erlbaum Associates.

Kozma, R. B. (1994), "Will Media Influence Learning? Reframing the Debate." Educational Technology Research & Development, 42(2), 7-19.

Leake, D. (1996), "CBR in Context: The Present and Future." In D. Leake (Ed.), Case-based Reasoning: Experiences, Lessons and Future Directions (pp. 3-30). Menlo Park, CA: AAAI Press/MIT Press.

Mitchell, T. M. (1997), Machine learning, New York: McGraw-Hill.

Raab, J., Rasala, R., & Proulx, V. K. (2000), "Pedagogical Power Tools for Teaching Java." Proceedings of the 5th Annual SIGCSE/SIGCUE ITiCSE Conference on Innovation and Technology in Computer Science Education, Helsinki, Finland, 156-159.

Roberts, E. (2004), "The Dream of a Common Language: The Search for Simplicity and Stability in Computer Science Education." Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education, Norfolk, Virginia, USA, 115-119.

Schank, R. (1982). Dynamic Memory: A Theory of Learning in Computers and People, Cambridge, England: Cambridge University Press.

Schank, R. C., & Cleary, C. (1995), Engines for Education, Retrieved June 10, 2004 from http://www.engines4ed.org/hyperbook/

Schmidt, C. (2006), "Towards the Support of Cognitive Flexibility in Hypertext Design." Journal of Computing Sciences in Colleges, Vol 21, Number 4

Schmidt, C. (2004), CBJava, Retrieved July 17, 2006 from http://192.104.1.46:8080/cbJava/. Unpublished work.

Spiro, R. J., Coulson, R. L., Feltovich, P. J., & Anderson, D. K. (1988), "Cognitive Flexibility Theory: Advanced Knowledge Acquisition in Ill-structured Domains." (Technical Report No. 441). Champaign, IL: University of Illinois at Urbana-Champaign.

Spiro, R. J., Feltovich, R. P., Jacobson, M. J., & Coulson, R. L. (1992), "Cognitive Flexibility, Constructivism, and Hypertext: Random Access Instruction for Advanced Knowledge Acquisition in Ill-structured Domains." In T. M. Duffy, & D. H. Jonassen (Eds.), Constructivism and the Technology of Instruction: A Conversation (pp. 57-76). Hillsdale, NJ: Erlbaum.

Spiro, R. J., & Jacobson, M. J. (1995), "Cognitive Flexibility, and the Transfer of Complex Knowledge: An Empirical Investigation." Journal of Educational Computing Research, *12*(4), 301-333.

AUTHOR BIOGRAPHY

Cecil Schmidt is an Associate Professor of Computer Information Sciences at Washburn University in Topeka, KS. Dr. Schmidt currently teaches courses in Computational Intelligence, Database, Data Mining, Object Oriented Programming, and Systems Analysis. His primary research interests are in the understanding of human learning and how machine intelligence might be developed and used to support human learning and understanding of complex topics.

Information Systems & Computing
Academic Professionals

Serving Information Systems Educators

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096