IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITB12726

This paper appears in the book, *Emerging Trends and Challenges in Information Technology Management, Volume 1 and Volume 2* edited by Mehdi Khosrow-Pour © 2006, Idea Group Inc.

Teaching Java: Applications of Programmed Instruction and Collaborative Peer Tutoring[1]

Henry H. Emurian, Information Systems Department, UMBC, 1000 Hilltop Circle, Baltimore, MD 21250,
T 410-455-3206, F 410-455-1073, emurian@umbc.edu

BACKGROUND

Our previous research (Emurian, 2005; 2006) showed that students who completed a Java tutoring system, which taught a simple ten-line Applet program to display a text string in a browser window, learned general rules of Java programming that could be applied to answer questions on problems not explicitly presented in the tutor text itself. These findings supported the value of the tutor to produce *meaningful learning* (Mayer, 2002) or *far transfer of learning* (Barnett & Ceci, 2002). The research methodology is similar to design-based research (Design-Based Research Collective, 2003; Hoadley, 2004) in that instructional design effectiveness was assessed iteratively over several successive semesters within the context of the classroom.

To potentiate the outcomes associated with our students' use of the tutoring system, collaborative peer tutoring was considered as an additional tactic to facilitate learning. Collaborative peer tutoring is a social situation in which students teach, coach, and/or evaluate each other within groups of two or more students. This social dimension may enhance motivation to learn and may provide the occasion for mutual elaboration of the understanding of simple facts and concepts to exceed what might be accomplished by solitary study (Rittschof & Griffin, 2001; Slavin, 1996). The potential benefits of peer collaboration when applied to learning computer programming have been explored in the learning of recursive programming using LISP-LOGO (Jehng, 1997), and in a pair programming laboratory for introductory Java (Williams et al., 2002). Among the variants of collaborative peer tutoring paradigms, this study adopts *interteaching* (Boyce & Hineline, 2002), operationalized by having two students engage in a face-to-face dialog to discuss questions previously given to them.

The research reported here is based upon two successive offerings of an elective course entitled *Graphical User Interface Systems Using Java*. The first class, offered during the summer of 2004, consisted of master's degree students, and the second class, offered during the fall of 2004, consisted of advanced undergraduate students.

METHOD

Materials

The questionnaires, interteaching report (also showing the program to be learned), and other material are available on the web[2] as PDF documents. Access to the web-based Java tutor is also available[3], and the source code is freely available.

Subjects

There were 14 students in each class. Background data was collected during the first questionnaire administration (Pre-Tutor Questionnaire). The summer (S) 2004 class had 6 female and 8 male students, and the fall (F) 2004 class had 1 female and 13 male students (chi-square = 4.76, df = 1, p < .05).

Figure 1.

	Summer 2004 Master's Degree Students n = 14	Fall 2004 Advanced Undergraduates n = 14
Sessions 2.5 Hours	**Tutor** Questionnaires: SSE & Rules	**Tutor + Interteaching** Questionnaires: SSE, Items, Lines, & Rules
1	1. Pre-Tutor Questionnaires 2. Tutor 3. Post-Tutor Questionnaires	1. Pre-Tutor Questionnaires 2. Tutor
Homework		*Access to Study Manual*
2	1. Lecture 2. Run the Program 3. Final Questionnaires	1. Post-Tutor Questionnaires 2. Interteaching 3. Lecture 4. Run the Program
3		1. Final Questionnaires • Test Credit for Items, Lines, & Rules

Experience ratings were based on a ten-point scale where ***1 = Novice*** to ***10 = Expert***. Comparisons between classes were based on the Kruskal-Wallis test. Median ratings were as follows: Java experience (S = 3, F = 1.5; chi-square = 3.51, p > .05), programming experience (S = 4, F = 5.5; chi-square = 3.34, p > .05), and number of programming courses taken (S = 3, F = 5; chi-square = 2.31, p > .10). The number of students reporting courses taken was only 13 for the fall 2004 class. The median age of the students was 26 years for S (range = 22 – 45) and 25 years for F (range = 20 – 32) (chi-square = 1.85, p > .10).

Procedure

Figure 1 presents the sequence of events for each of the two classes. The summer 2004 class met twice each week for six weeks. The fall 2004 class met once each week for 14 weeks. All classes met for 2.5 hours. The students were fully informed about the requirements of each class, and the sequence of events was included on the syllabus. All students completed the tutor during Session 1, the first class period. After Session 1 for the fall 2004 students, a study manual was released that duplicated the instructional text within the tutor but that omitted the embedded multiple choice tests on the items and lines of code. Students were informed that the manual could be used to prepare for the interteaching on Session 2. The study manual and the interteaching report presented the 12 rules multiple-choice questions, but the answers were not given.

The sequence of events was a compromise that allowed student behavior to be evaluated within the context of an actual classroom. The justification for such a design-based research approach, together with its strengths and limitations, will be discussed.

RESULTS

On the figures to follow, the first class (summer 2004) is identified as Programmed Instruction (**PI**), which reflects a behavior analytic approach to technology education (Greer, 2002, p. 187). The second class (fall 2004) is identified as Programmed Instruction plus Interteaching (**PI + Inter**).

Figure 2.

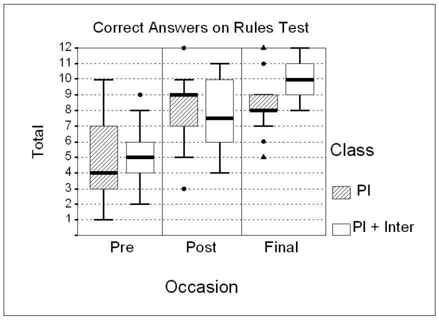


Figure 3.

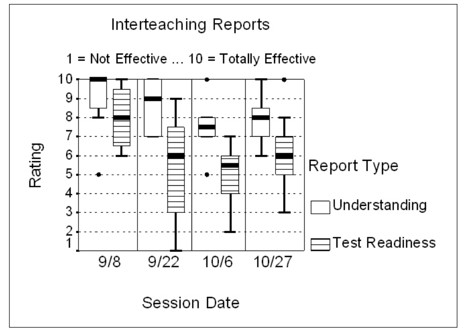


Figure 2 presents boxplots of total correct answers on the rules test across the three occasions for both classes. Kruskal-Wallis tests showed no significant difference between the classes for the Pre test (chi-square = 0.24, df = 1, p > .50) and the Post test (chi-square = 0.00, df = 1, p > .50). The difference for the Final test was significant (chi-square = 8.40, df = 1, p < .005).

Figure 3 presents boxplots of ratings of the effectiveness of the interteaching for the two types of ratings: (1) effectiveness of the dialog in understanding the material, and (2) effectiveness of the dialog in preparing for a test. The figure presents median ratings across the four sessions for the eight students who were present on all four occasions of interteaching. For understanding, the figure shows graphically that the highest median rating was observed on the first session, in which the value was the maximum of 10. Medians declined thereafter over the next two sessions, and the median increased somewhat during the fourth session. A Friedman's test, however, was not significant (chi-square = 5.10, df = 3, p > .15), indicating insufficient evidence to conclude that the changes in medians observed graphically were significantly different from each other. For test preparation, the figure shows graphically that the highest median rating was observed on the first session, and ratings were comparatively lower on the other three sessions. A Friedman's test was significant (chi-square = 11.11, df = 3, p < .05). Figure 3 also shows graphically that the median understanding rating was higher than the corresponding test readiness median across all four sessions. A Kruskal-Wallis comparison of the differences between the understanding and test readiness ratings, for all subjects across the four sessions, with a population of zeros was significant (chi-square = 33.75, df = 1, p <.001). The correlation between the two sets of ratings was not significant (r = 0.28, p > .10).

Figure 4 presents boxplots of software self-efficacy ratings across the three occasions for both classes. The ratings are based on the median confidence rating for all 21 unique items of code in the program. The figure shows graphically that students in both classes reported robust increases in confidence between the Pre and Post occasions, and the median rating reached the ceiling of ten on the Final occasion. For the PI class, Cronbach's alphas for Pre, Post, and Final occasions were 0.98, 0.99, and.0.99, respectively. The Final alpha was not significant. A Friedman's test was significant (chi-square = 23.24, df = 2, p < .001). For the PI+Inter class, Cronbach's alphas for Pre, Post, and Final occasions were 0.98, 0.98, and 0.97, respectively. All were significant. A Friedman's test was significant (chi-square = 20.49, df = 2, p < .001). Kruskal-Wallis tests of median ratings between the two classes were not significant for Pre (chi-square = 0.35, df = 1, p > .50), Post (chi-square = 2.54, df = 1, p > .10), and Final (chi-square = 2.16, df = 1, p > .10) occasions.

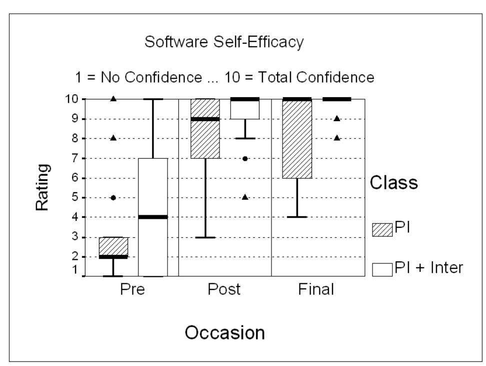
During the Final occasion, students rated the tutor along the following three dimensions: (1) overall impression, (2) effectiveness of the tutor in learning Java, and (3) usability of the tutor, where ***1 = Totally negative*** to ***10 = Totally positive***. The generally positive evaluation of the tutor is indicated by the fact that all medians were eight or higher. Kruskal-Wallis tests between the two classes were not significant for overall (chi-square = 0.22, df = 1, p > .50), learning Java (chi-square = 0.04, df = 1, p > .50), and usability (chi-square = .03, df = 1, p > .50) evaluations.

DISCUSSION

Students in two classes showed gains in program understanding and software self-efficacy as a function of participation in two sets of instructional experiences that were designed to facilitate learning a Java computer program. A programmed instruction tutoring system was effective in promoting student confidence and learning, and an interteaching dialog also contributed to performance when these tactics were used within the context of a classroom. Collaborative peer tutoring may have potentiated a student's understanding of general principles of Java intended to be taught by the individualized tutoring system. These outcomes show how several instructional tactics, along with a traditional lecture, may be managed in the classroom to the benefit of introductory programming students.

The present study falls within the scope of design-based research. This is an attempt to engineer an instructional environment by applying principle-based interventions to the classroom and by collecting data on learning effectiveness, while at the same time acknowledging that an actual classroom intervention introduces multiple sources of confounding variables that make causal attribution problematic (Brown, 1992; Edelson, 2002). As a type of formative evaluation (Collins, Joseph, & Bielaczyc, 2004), the essence of design-based research is systematic replication (Sidman, 1960) in the classroom. Improvements to a previously established and meritorious instructional approach are introduced and evaluated iteratively across successive offerings of a course. Theory informs the design, and the evaluations stimulate theoretical revisions and subsequent design alterations. The ultimate aim is to provide instructional experiences so that each and every student may reach an equivalent achievement objective.

Figure 4.



REFERENCES

Barnett, S.M., & Ceci, S.J. (2002). When and where do we apply what we learn? A taxonomy for far transfer. *Psychological Bulletin*, *128*, 612-637.

Brown, A.L. (1992). Design experiments: Theoretical and methodological challenges in creating complex interventions in classroom settings. *The Journal of the Learning Sciences*, *2*(2), 141-178.

Boyce, T.E., & Hineline, P.N. (2002). Interteaching: A strategy for enhancing the user-friendliness of behavioral arrangements in the college classroom. *The Behavior Analyst*, *25*, 215-226.

Collins, A., Joseph, D., & Bielaczyc, K. (2004). Design research: Theoretical and methodological issues. *The Journal of the Learning Sciences*, *13*(1), 15-42.

Design-Based Research Collective. (2003). *Educational Researcher*, *32*(1), 5-8.

Edelson, D.C. (2002). Design research: What we learn when we engage in design. *The Journal of the Learning Sciences*, *11*(1), 105-121.

Emurian, H.H. (2005). Web-based programmed instruction: Evidence of rule-governed learning. *Computers in Human Behavior*. *21*(6), 893-915.

Emurian, H.H. (2006). A web-based tutor for Java™ Evidence of meaningful learning. *International Journal of Distance Education Technologie*s, in press.

Greer, R.D. (2002). *Designing teaching strategies: An applied behavior analysis systems approach*. NY: Academic Press.

Hoadley, C.M. (2004). Methodological alignment in design-based research. *Educational Psychologist*, *39*(4), 203-212.

Jehng, J-C. J. (1997). The psycho-social processes and cognitive effects of peer-based collaborative interactions with computers. *Journal of Educational Computing Research*, *17*(1), 19-46.

Mayer, R.E. (2002). *The promise of educational psychology. Volume II. Teaching for meaningful learning*. Upper Saddle River, NJ: Pearson Education, Inc.

Rittschof, K.A., & Griffin, B.W. (2001). Reciprocal peer tutoring: Re-examining the value of a co-operative learning technique to college students and instructors. *Educational Psychology*, *21*(3), 313-331.

Sidman, M. (1960). *Tactics of scientific research*. New York: Basic Books.

Slavin, R.E. (1996). Research on cooperative learning and achievement: What we know, what we need to know. *Contemporary Educational Psychology*, *21*, 43-69.

Williams, L.A., Wiebe, E., Yang, K., Ferzli, M., & Miller, C. (2002). In support of pair programming in the introductory computer science course. *Computer Science Education*, *12*(3), 197-212.

ENDNOTES

[1] An extended version of this paper will appear in the International Journal of Information & Communication Technology Education.

[2] http://userpages.umbc.edu/~emurian/learnJava/irma2006/

[3] http://nasa1.ifsm.umbc.edu/learnJava/tutorLinks/TutorLinks.html

Related Content

Promotion of Administrative Modernization through Processes Dematerialization
Liliana Ávila, Leonor Teixeiraand Pedro Almeida (2015). *Encyclopedia of Information Science and Technology, Third Edition (pp. 640-649).*
www.irma-international.org/chapter/promotion-of-administrative-modernization-through-processes-dematerialization/112377

Fuzzy Decoupling Energy Efficiency Optimization Algorithm in Cloud Computing Environment
Xiaohong Wang (2021). *International Journal of Information Technologies and Systems Approach (pp. 52-69).*
www.irma-international.org/article/fuzzy-decoupling-energy-efficiency-optimization-algorithm-in-cloud-computing-environment/278710

Ecosystem Wetlands Restoration Approach for Sustainable Development Planning
Carolina Collaro (2015). *Encyclopedia of Information Science and Technology, Third Edition (pp. 2931-2941).*
www.irma-international.org/chapter/ecosystem-wetlands-restoration-approach-for-sustainable-development-planning/112716

New Trends and Tools for Customer Relationship: Challenges in Digital Transformation
Ana Limaand Jorge Pacheco (2019). *Educational and Social Dimensions of Digital Transformation in Organizations (pp. 1-26).*
www.irma-international.org/chapter/new-trends-and-tools-for-customer-relationship/215133

Using Technology to Connect Students with Emotional Disabilities to General Education
Alicia Roberts Frank (2013). *Cases on Emerging Information Technology Research and Applications (pp. 349-362).*
www.irma-international.org/chapter/using-technology-connect-students-emotional/75868