Collaborative Example Authoring System: The Value of Re-annotation based on Community Feedback

I-Han Hsiao, Peter Brusilovsky
School of Information Sciences, University of Pittsburgh
135 N. Bellefield Ave., Pittsburgh, PA 15260 USA
{ihh4, peterb}@pitt.edu

Abstract. Learning from examples is a common and powerful approach when mastering the art of programming. In our classroom studies of WebEx and NavEx, students highly praised the systems. However, the broader dissemination of this approach was not very successful due to the lack of content – annotated examples. This paper presents the study results in example-based programming learning by annotating examples. The classroom study confirmed that community successfully filtered out good and bad annotations and improved the quality of the annotations. In addition, the annotating example assignment was perceived highly helpful in understanding.

1. Introduction

Learning from examples is a common and powerful approach when mastering the art of programming. It encourages students to reuse the code of previously analyzed examples in solving a new problem (Brna 1999; Weber & Brusilovsky 2001). Gomez-Albarran (2005) in a synthesis report about teaching and learning of programming stressed that example-based learning is a natural way of learning. To support online learning from examples in programming courses we developed WebEx (Web Examples) System. WebEx providing interactive access to examples enhanced with line-by-line comments (Brusilovsky 2001). It allows students to browse the comments at their own pace and order (Sosnovsky, Brusilovsky & Yuldelson 2004). NavEx (Navigation to Examples) was presented in 2004 to provide adaptive navigation support (Brusilovsky 2004).

In our classroom studies of WebEx and NavEx, students highly praised the systems. However, the broader dissemination of this approach was not very successful due to the lack of content – annotated examples. Teachers usually have limited time to annotate the huge amount of examples. Examples are simply so many, but annotations are so few. This paper explores the feasibility of an alternative authoring approach - community-based development of annotations.

In previous survey of NavEx, 70% of the participants responded that they would like to be able to create their own

annotated examples or add their own annotations to the code lines (Yudelson & Brusilovsky 2005). Moreover, in the context of a programming course, authoring (rather than only using) examples could be considered as a useful learning activity. Jonassen and Reeves (1998) contend that students are likely to learn more by constructing hypermedia instructional materials that by studying hypermedia created by others. The question that we had to answer is whether the students will be able to create quality content – something, which can be used by others as a learning resource? One of the modern approaches to ensure a quality of content in educational repositories is a peer review mechanism, which harness the power of the community of users. This lead to the second question: will the peer-review mechanism work in a community of students?

This paper reports a study, which attempted to answer these questions. Therefore, this study firstly aimed to solicit the example annotations from the student community. Following this, the collected annotations were passed through peer review upon the community. Moreover, in terms of quality control, students were allowed to re-annotate the badly written annotations based on the comments provided from the community. The final results were carefully examined through review by domain experts. Succinctly put, the study confirmed that the learning community was successfully able to discern between good and bad annotations and improved the quality of the annotations.

2. Related Work

According to Chi and her colleagues (1989), students can learn a lot when attempting to explain examples. Other cognitive science studies have shown that students acquired less shallow procedural knowledge by specifically giving an explanation (Aleven & Koedinger 2002). Chi et al. (1989) showed that self-explanations in the context of learning about mechanics from worked-out examples had rather dramatic effects on participants' ability to solve problems on their own. Hence, we hypothesized that assigning annotation-writing to explain lines of program code might increase the students' knowledge of the programming language. Also, commenting on the annotations is considered as additional form of explanation activity, refining the articulation of the example program's content.

Calibrated Peer Review (CPR) System supports student learning by giving them writing assignments about important course topics (Chapman 2001). Through the peer review process, students will be able to learn to read for content. At the same time, it's an exercise to develop reviewing skills. In the broader sense of education implication, perceived helpfulness is likely to mediate between the feedback and the revisions made in later writing (Rucker & Thompson 2003). Furthermore, peer assessment and numeric ratings are commonly used to analyze the validity and reliability (Cho & Schunn 2003; 2006). Therefore, we also used peer review technique to examine the annotation quality, as judged by the community.

3. Collaborative Example Authoring System

Collaborative Example Authoring System (http://kt1.exp.sis.pitt.edu:8080/example/) was previously used as an authoring tool allowing teachers to create programming example codes and comments. However, in order to observe the value of student re-annotation of programming examples, based on community feedback, the system was refined and opened up to the students. Students are able to annotate the examples, provide comments on annotations, and give ratings for the example annotations. Figure 1 to 4 are the snapshots of the system interfaces.

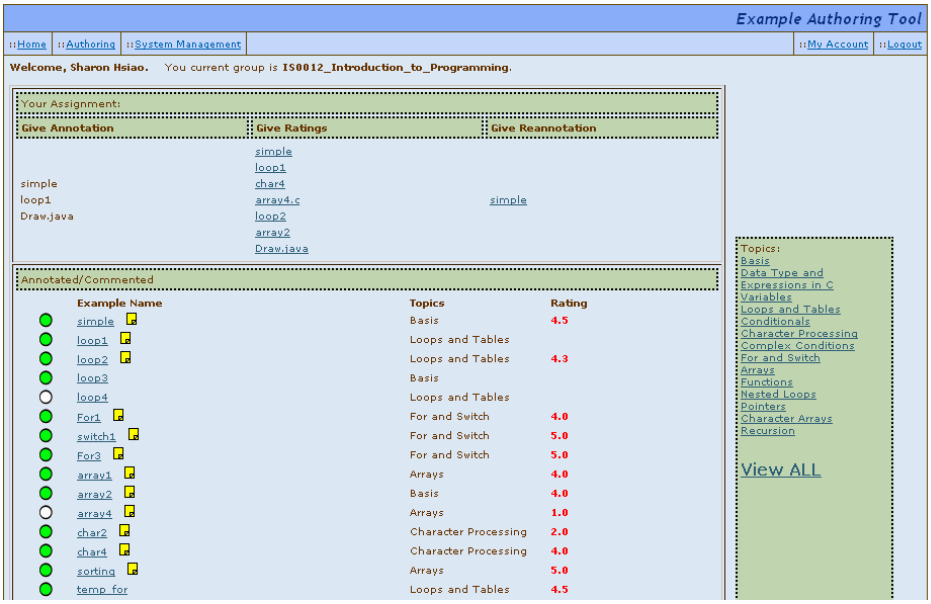

Figure 1: System Interface: Collaborative Example Authoring System

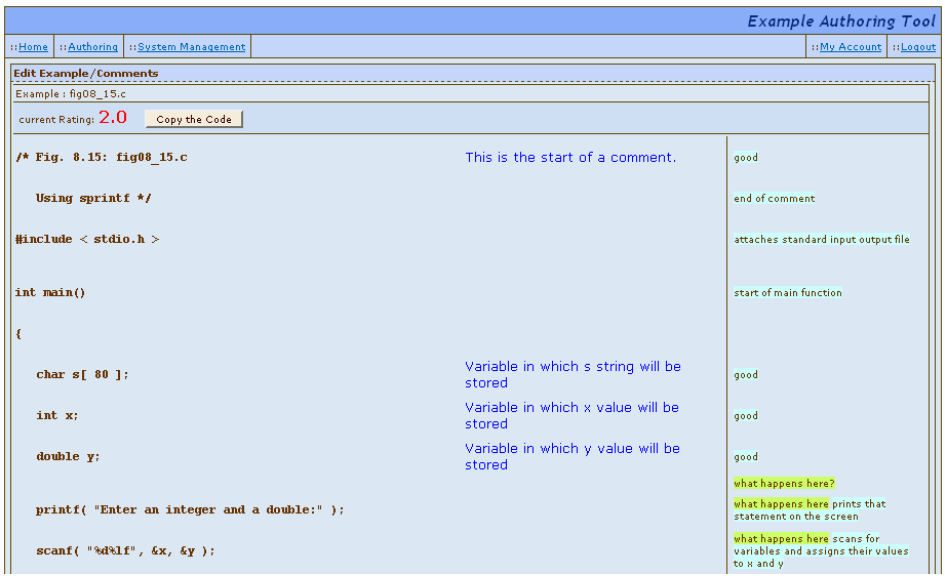

Figure 2: System Interface: View an Annotated Example

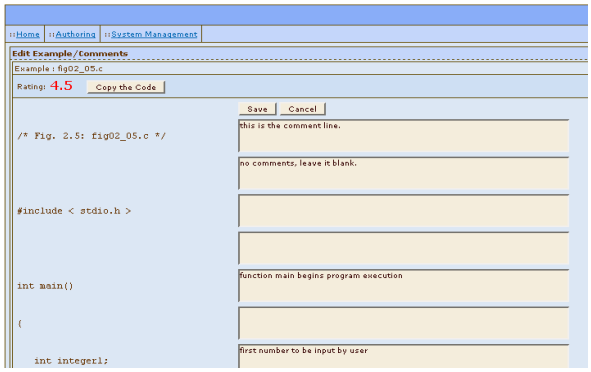

Figure 3: System Interface: Annotating an Example

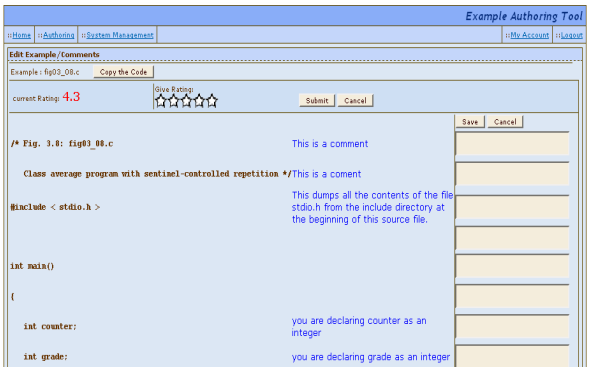

Figure 4: System Interface: Rate/Comment the Annotations

4. Hypotheses

To investigate the impact of peer reviewing and renovation process supported by the Collaborative Example Authoring System, we run an empirical study. The study attempted to check the following hypotheses.

1. The community will filter out good and bad annotations through giving ratings and comments about the annotations. High ratings represent better quality; low ratings represent worse quality.

2. Re-annotation improves the quality.

3. Adding annotation helps the student understand program examples.

5. Study Design

In order to observe the value of re-annotation, the study was divided into three phases, each lasting three weeks. Each phase contains different tasks which are graded separately, as part of their weekly assignment. The detail will be addressed in the later section of this paper. The subjects are students from an Introduction to Programming course offered by the School of Information Sciences, at the University of Pittsburgh. There are seven undergraduate students in total. The programming examples are culled from the textbooks in Knowledge Sea II System (Brusilovsky, Chavan & Farzan 2004), a mixed corpus C programming resource, which is also available to students online within the supplemental course materials section of the course. The example topics selected in this study respectively were: conditional, variable, loops, for, switch, character processing and array. They had all been covered in previous lectures.

First Phase (Annotating): Each student had to give annotations on two C programming examples. The topics were randomly assigned.

Second Phase (Rating and Commenting): The annotated examples were collected. Each student was asked to provide ratings to six annotated examples. A five-star rating technique was used; one to five stars represent a strongly negative to a strongly positive continuum. However, providing comments about the annotations was optional.

Third Phase (Re-annotating): The ratings and comments were collected. According to the community ratings, we categorized the examples into two groups, a high ratings and a low ratings group. The low ratings group of examples was reassigned back to students randomly. Based on the community ratings and comments, students were allowed to provide or change annotations on the example assigned.

During the experiment, students were able to see all the examples from the community pool but anonymous authors; however, they were only allowed to modify the examples assigned to them. Furthermore, after completing the authoring the annotations in the first phase, the system locked the examples assigned to them, making sure they could not go back and modify the content until after the rating/commenting phase was completed.

6. Results

After the completion of these three stages, all of the annotated examples were firstly passed a context quality examination by Expert Review. Each given annotation was rated. The correlation of before re-annotation ratings between community and experts is high (r=0.95). It proves that the community successfully distinguished good and bad annotations.

	High Ratings Group	Low Ratings Group	Low Ratings Group(Re-annotated)
Annotating rate	**69.5%**	**50.9%**	**59.2%**
Community Average Ratings	**4.86**	**3.40**	
Expert Ratings	**4.68**	**3.35**	**4.18**
Comments			
Praise/agreement	**91**	**48**	
Supplemental annotations	**2**	**61**	
Questions	**0**	**5**	

Table 1: Summary of High and Low Ratings Groups

After re-annotation, the average ratings of the low rating group were increased (Tab. 1). In each example, the correlation between before and after re-annotation was generally high. Fig. 5 indicates the improvement of each example after the re-annotation process. In Fig. 5, example no#6, the ratings doubled after re-annotation, but the correlation was relatively low. This is due to the fact that this specific example's annotating rate (20.00%) was low before re-annotation. In other words, the given comments from the community at phase2 and additional annotations provided through phase3 contributed to higher ratings. Overall, each example received higher ratings after re-annotation. Consequently, the annotation quality was improved after the re-annotation process. Meanwhile, the average ratings of high ratings group were 4.86 from the community, 4.68 from the expert review. Although there's a minor difference on the average points for these examples, it shows that the community tends to provide slightly higher ratings than experts do.

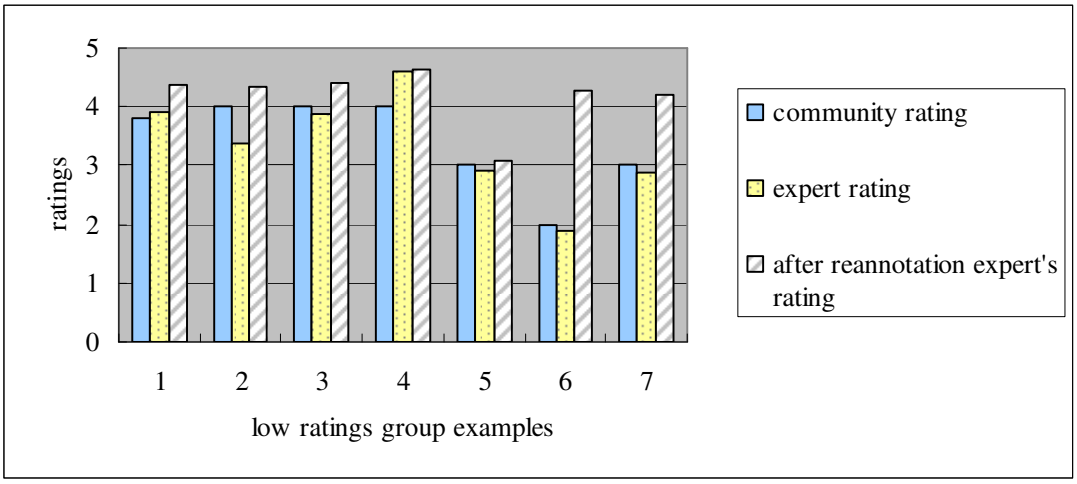

Figure 5: Comparing community rating, expert rating and after re-annotation expert rating for annotated examples in low ratings group.

There are other interesting findings from the data. Firstly, in the high ratings group, there are only 2 supplemental annotations given while students were at the second phase. On the other hand, 61 supplemental annotations were provided in the low ratings group. It's 71% of the total annotation lines. Moreover, 5 questions were raised; specifically asking "what happened here?" It indicates the insufficiency of the annotations in the low ratings group. It's noteworthy that all of the 5 questions were in the lowest rating example. However, they were answered and explained by annotation during the third phase.

Secondly, there are 91 of the comments specifically praised or said "ok" in agreement with the annotations in the high ratings group. On the other hand, the low ratings group, praise and agreement with the annotation were about half of the amounts given to the high ratings group annotations. Again, this supports our first hypothesis that community is able to filter out good and bad annotations.

Last but not least, after the re-annotation process, 14 new annotations were given. The annotating rate climbs a little after re-annotation (Tab. 1). It is noteworthy that 52.3% of the annotations were changed. We categorized them into four types of the re-annotation: re-annotations modified from original annotations, modified from comments, exactly the same as comments and completely new re-annotations (Fig. 6). In order to see the quality of improvement for each annotation, we correlate the modification to ratings change. The correlation between re-annotations based on community (sum of first three types) and improved quality is high (r=0.927). However, the correlation between completely new re-annotations and improved quality is only 0.092. Thus, no matter what the changes were, whether arising from the original annotations or due to comments, they ultimately resulted in ratings increases. The completely new annotations might also be affected by community ratings; however, the new ones did not pass through community examination again. Therefore the quality does not seem to be as good as the other types of re-annotation.

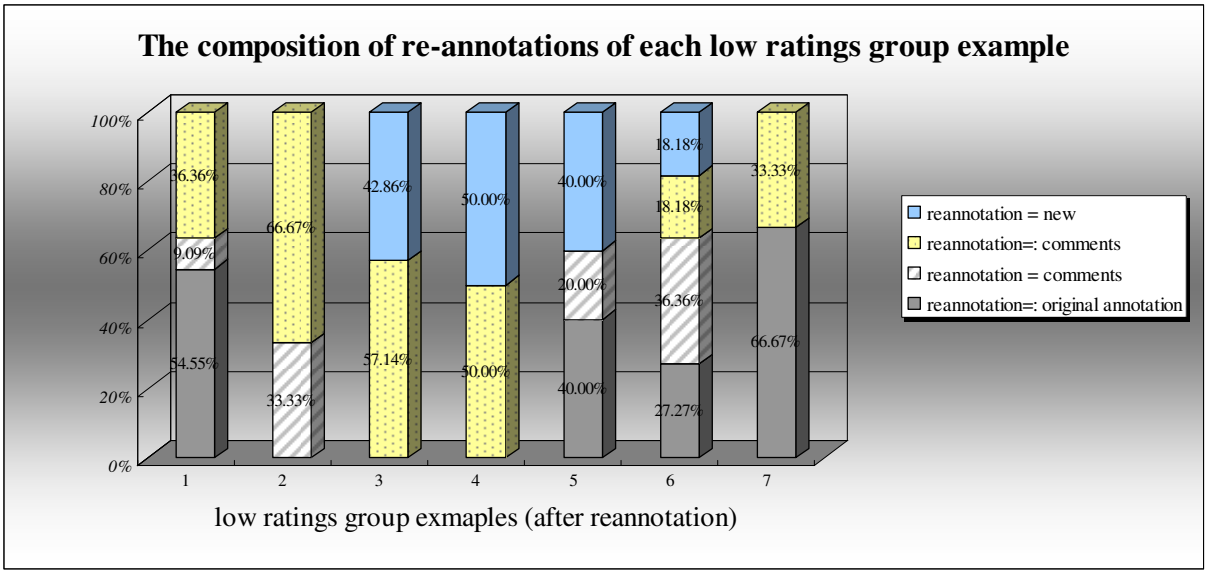

Figure 6: The composition of re-annotations of each low ratings group example

7. Subjective Data Analysis

In addition, a non-mandatory questionnaire was administered at the end of the experiment. Students' opinions and suggestions were collected by asking questions in regard to key features of the system. 5 out of 7 students completed the questionnaire. As you can see from Fig. 7, 80% of the students felt positive or strongly positive about the need for such tool in general, as well as describing how it helped in their understanding. More than 90% of the students found it useful and that it complied with the scope of this learning activity. Students found that the task of annotating examples improved their knowledge of programming skills. Besides, annotating examples as homework was a good exercise for practicing and reflecting what was taught in class. 100% of the students positively or strongly positively

liked the system interface itself. There's no single negative opinion in this survey.

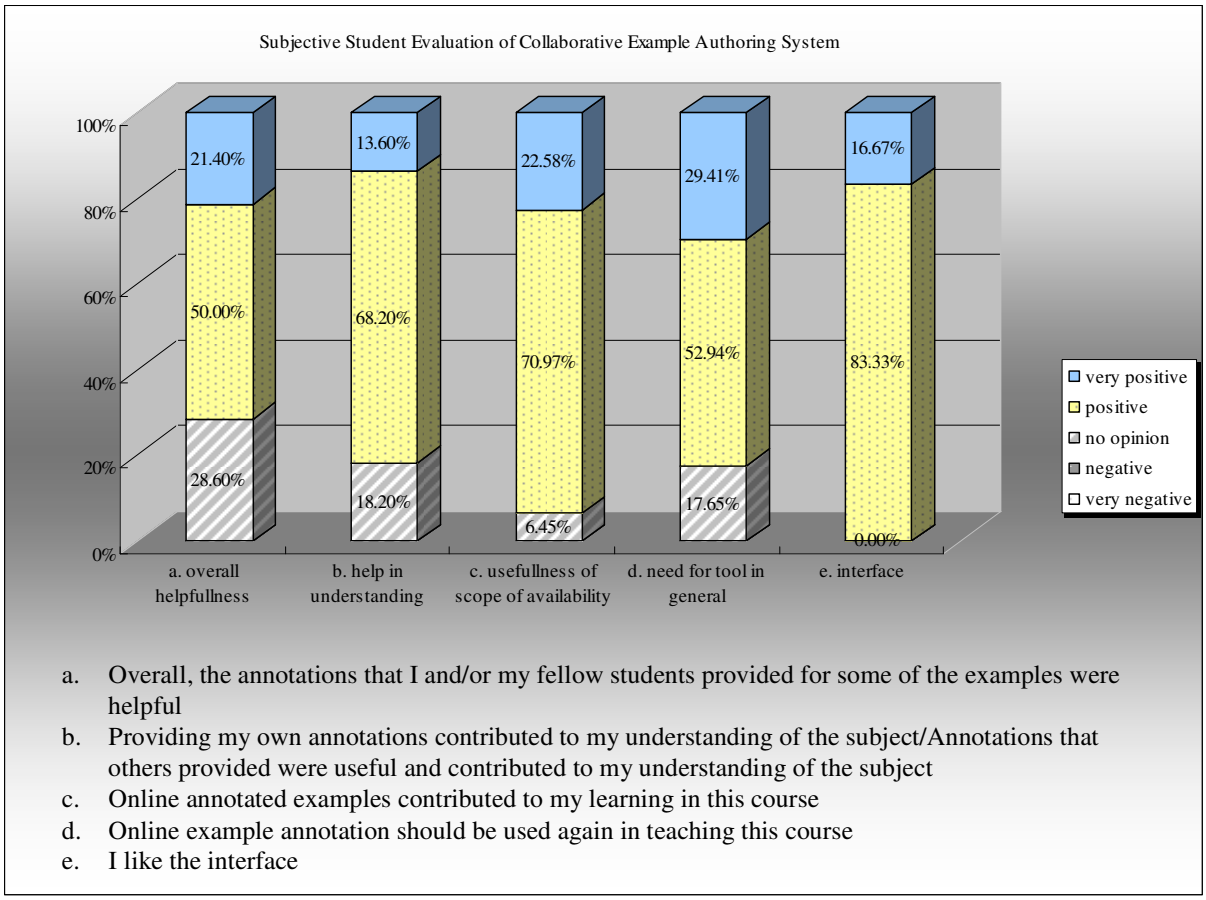

Figure 7: Subjective Student Evaluation of Collaborative Example Authoring System

We also conducted another qualitative subject study with two graduate students. The think-aloud process was observed while they were completing the tasks through its three phases. Both cases began by giving annotations immediately right after a quick read through the codes. If the students bumped into uncertainties, they searched online help or had discussions with others to clarify their understanding. When almost finished with the annotations, they copied the code, then actually compiled it and ran it on their local machines. By seeing the execution results, they went back to their screens and gave more annotation. While one of them was in the process of annotating a "for loop" section, she slowed down and read carefully through the codes, then specifically said "hmm, very cool. It would be a helpful tool for learning programming language". Additionally, both of them found this annotating assignment to be helpful to their understanding.

8. Summary and Future Work

The results are consistent with the hypotheses. The community successfully gave ratings and comments to indicate good or bad annotations. In addition, the low ratings group's annotations were improved after the re-annotation, which was again based on the community feedback. Also, perceived usefulness and helpfulness were statistically high. Thus, as far as programming language learning concerned, the results explicitly encourage the use of the Collaborative Example Authoring System as an educational tool

Due to the fact that the Collaborative Example Authoring System was originally designed for authors/teachers creating examples for programming language learning, it would be interesting to see whether students would also be able to create valuable examples, as well as giving annotations. To back this up, there is a big chunk of research regarding to self-explanation promotes understanding (Chi 1996). Thus, in order to improve students' understanding in programming language learning, it would be beneficial to use eye-tracking devices as well as observe students' exploratory behavior directly, in hopes of collecting enough quantitative information to show the link to improved understanding. A further study with a larger student population has already been scheduled for this summer. From this summer's study, we expect a more representative sampling of comments on annotations and increasingly significant results.

Acknowledgement

This research is supported by the National Science Foundation under Grant No. 0447083

References

Aleven, V. & Koedinger, K.(2002) An effective metacognitive strategy: learning by doing and explaining with a computer-based Cognitive Tutor*, Cognitive Science: A Multidisciplinary Journal*, 26(2), 147-179

Brna, P. (1999) Searching for Examples with a Programming Techniques Editor, *Journal of Computing and Information Technology,* 6 (1), 13-26.

Brusilovsky, P. (2001) WebEx: Learning from examples in a programming course. In: W. Fowler and J. Hasebrook (eds.) Proceedings of WebNet'2001, World Conference of the WWW and Internet, Orlando, FL, October 23-27, 2001, AACE, pp. 124-129.

Brusilovsky, P., Chavan, G., and Farzan, R., (2004) Social Adaptive Navigation Support for Open Corpus Electronic Textbooks. In Nejdl, Wolfgang and De Bra, Paul (Eds.) Proceedings of the Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, (pp. 24-33). Eindhoven, The Netherlands: Springer LNCS 2004.

Chapman, O. (2001) Calibrated Peer Review (CPR) White Paper, available at http://cpr.molsci.ucla.edu/

Chi, M. T. H., Bassok, M., Lewis, M., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, 13, 145-182.

Chi, M.T.H. (1996) Constructing self-explanations and scaffolded explanations in tutoring. Applied Cognitive Psychology, 10, pp. S33-S49.

Cho, K., & Schunn, C. D. (2003). Validity and reliability of peer assessments with a missing data estimation technique. Proceedings of ED-Media 2003, 1511-1514.

Cho, K., & Schunn, C. D. (2006) Commenting on Writing: Typology and Perceived Helpfulness of Comments from Novice Peer Reviewers and Subject Matter Experts, Written Communication, 23(3), pp 260-294

Gomez-Albarron, M., (2005), The Teaching and Learning of Programming: A Survey of Supporting Software Tools, The Computer Journal, Volume 48, Issue 2, March 2005: pp. 130-144

Jonassen, D. H. & Reeves, T. C. (1996). Learning with technology: Using computers as cognitive tools. In Jonassen, D. H. (Ed.), Handbook of Research for Educational Communications and Technology (pp. 693-719). New York, NY: Mc Millan.

Rucker, M. L., & Thompson, S. (2003). Assessing student learning outcomes: An investigation of the relationship among feedback measures. College Student Journal, 37, 400-404.

Weber, G. and Brusilovsky, P. (2001) ELM-ART: An Adaptive Versatile System for Web-based Instruction, International Journal of Artificial Intelligence in Education (2001), 12, 351-384

Yudelson, M. and Brusilovsky, P. (2005) NavEx: Providing Navigation Support for Adaptive Browsing of Annotated Code Examples. In: C.-K. Looi, G. McCalla, B. Bredeweg and J. Breuker (eds.) Proceedings of 12th International Conference on Artificial Intelligence in Education, AIED 2005, (Amsterdam, July 18-22, 2005). Amsterdam: IOS Press, pp. 710-717.