

Adaptive Navigation Support for Parameterized Questions in Object-Oriented Programming

I-Han Hsiao, Sergey Sosnovsky, Peter Brusilovsky

School of Information Sciences, University of Pittsburgh, USA
{ ihh4,sas15,peterb}@pitt.edu

Abstract. This paper explores the impact of adaptive navigation support on student work with parameterized questions in the domain of object-oriented programming. In the past, we developed QuizJET system, which is able to generate and assess parameterized Java programming questions. More recently, we developed JavaGuide system, which enhances QuizJET questions with adaptive navigation support. This system introduces QuizJET and JavaGuide and reports the results of classroom studies, which explored the impact of these systems and assessed an added value of adaptive navigation support. The results of the studies indicate that adaptive navigation support encourages students use parameterized questions more extensively. Students are also 2.5 times more likely to answer parameterized questions correctly with adaptive navigation support than without such support. In addition, we found that adaptive navigation support especially benefit weaker students helping to close the gap between strong and weak students.

Keywords: adaptive navigation support, parameterized quizzes, self-assessment, object-oriented programming.

1 Introduction

Parameterized questions and exercises [1] emerged as an active research area in the field of E-Learning . This technology allows generating many objective questions from a relatively small number of templates created by content authors. Using randomly generated parameters, every question template is able to produce many similar, yet sufficiently different questions. As demonstrated by a number of projects such as CAPA [2], WebAssign [3], EEAP282 [4], Mallard [5], parameterized questions can be used effectively in a number of domains allowing to increase the number of assessment items, decrease authoring efforts, and reduce cheating.

The work of our research group focused on exploring the value of parameterized questions in the area of computer programming. We have developed and explored QuizPACK [1], a system which is able to generate and assess parameterized questions for C programming. Unlike the majority of modern system for automatic assessment of programming exercises [6-8], which focus on program-writing exercises, QuizPACK focused on program-tracing questions. This kind of questions is known as very important [9], however there are almost no question generation and assessment

systems, which can work with these questions. QuizPACK was evaluated in a series of classroom studies, which confirmed the educational value of this technology [1, 10]. We also found that the parameterized questions can be successfully combined with the technology of adaptive navigation support. The use of adaptive navigation support to guide students to most appropriate questions was found to increase student ability to answer questions correctly and encourage them to use the system more extensively (which, in turn, positively impacted their knowledge) [11].

While we confirmed the value of adaptive navigation support for parameterized questions in several studies, our earlier research left a number of questions unanswered. First, parameterized questions in area of C programming were not as diverse from the complexity point of view as questions in other areas such as physics [2]. As a result, it was left unclear whether adaptive navigation support can work in the context of a broader range of question difficulty from relatively simple to very difficult. Second, due to the decreased number of students in C programming classes, we were not able to separately assess the impact of this intelligent guidance technology on stronger and weaker students, which is an important typical research question.

To answer these questions, we expanded our work on parameterized questions to a more sophisticated domain of object-oriented Java programming, which is now the language of choice in most introductory programming classes. This domain allows us both: to introduce questions of much broader complexity and to explore our ideas in larger classes. Capitalizing on our experiences with QuizPACK, we developed QuizJET (Java Evaluation Toolkit), which supports authoring, delivery, and evaluation of parameterized questions for Java [12]. A preliminary evaluation has demonstrated that parameterized questions work really well in this domain: we found a significant relationship between the amount of work done by students in QuizJET and their performance. Working with QuizJET, students were able to improve their in-class weekly quiz scores. We also found that their success in QuizJET (higher percentage of correct answer) correlates with high scores on the final exam.

This paper presents the result of our second study of parameterized questions for object-oriented Java programming. The goal of this study was to assess the added value of adaptive navigation support on the student work with parameterized questions in this domain. In addition to exploring this technology combination in a new domain, the study specifically attempted to assess the impact of adaptive navigation support to student work with questions of different complexity as well as the impact of this technology on weaker and stronger students.

In this study we compared the impact of parameterized questions in two introductory Java programming classes, featured the instructor, syllabus, and student cohort (undergraduate information science students at the University of Pittsburgh). One of these classes used QuizJET system and another used JavaGuide system, which is QuizJET enhanced with adaptive navigation support service QuizGuide [13, 14]. In the following sections we present briefly QuizJET and JavaGuide systems and report the results of our classroom studies. We conclude with a summary of results and some discussion of future work.

2 QuizJET: Parameterized Questions for Object-Oriented Programming in Java

QuizJET system was developed to explore the technology of parameterized questions in a challenging domain of object-oriented programming. QuizJET supports authoring, delivery and evaluation of parameterized questions for Java programming language. It covers a broad range of Java topics from Java basics to such advanced topics as objects, classes, polymorphism, inheritance, and exceptions.

The image shows two screenshots of the QuizJET interface. The top screenshot displays a Java code snippet for a class named 'Tester' with a 'main' method. The code initializes a 'BankAccount' object with a balance of 49.0. It then checks if the balance is greater than 50.0. If true, it withdraws 50.0; otherwise, it deposits 50.0. Finally, it prints the balance to a variable 'result'. Below the code, the question asks for the final value of 'result', and there is a 'Submit' button.

```
public class Tester {
    public static void main(String[] args) {

        BankAccount myBankAccount = new BankAccount(49);
        if ( myBankAccount.getBalance() > 50 ) {
            myBankAccount.withdraw(50);
        }
        else {
            myBankAccount.deposit(50);
        }

        double result = myBankAccount.getBalance();
    }
}
```

What is the final value of **result**?

Submit

The bottom screenshot shows the same code snippet, but with the evaluation results. It indicates that the answer is correct and shows both the user's answer and the correct answer as 99.0. There is a 'Try Again' button.

```
public class Tester {
    public static void main(String[] args) {

        BankAccount myBankAccount = new BankAccount(49);
        if ( myBankAccount.getBalance() > 50 ) {
            myBankAccount.withdraw(50);
        }
        else {
            myBankAccount.deposit(50);
        }

        double result = myBankAccount.getBalance();
    }
}
```

What is the final value of **result**?

CORRECT!

Your Answer is:
99.0

Correct Answer is:
99.0

Try Again

Fig. 1. The presentation (top) and the evaluation results (bottom) of a QuizJET question.

The delivery component of QuizJET allows students to access each question pattern through a Web browser using a unique link (in original QuizJET, these links were included into the course portal). Once a link to the question is accessed, QuizJET generates a unique instantiation of the question (Figure 1), which features a small Java program. The student's challenge is to mentally execute the program and answer a question such as: "What will be the final value of the specific variable?" or "What will be printed in the console window?" A tabbed interface design supports straightforward access to the full code of the problem, one Java class per tab. The driver class named Tester Class, containing the main function, is presented on the first tab, while other tabs show supporting classes (such as BankAccount in Figure 1). The tabbed arrangement uniquely characterizes object-oriented programming where even simple object-oriented program may require one or more imported classes.

To answer a question, students fill the answer in the input field and hit *Submit*. The system immediately reports the evaluation results and the correct answer (Figure 1, bottom). Whether their results were correct or not, students can hit the Try Again button to assess the same question pattern with different parameters. This function helps students achieve mastery of the topic. While it looks relatively simple from user's point of view, this functionality is supported by sophisticated authoring (performed with a separate authoring component), generation, and evaluation mechanisms, which are described in [12].

3 JavaGuide: Adaptive Navigation Support for QuizJET Questions

JavaGuide is an adaptive hypermedia system, which provides adaptive navigation support for QuizJET questions. It is not a part of QuizJET, but an independent system, which simply hosts links to QuizJET questions and guides the students to most appropriate links using adaptive link annotation. In this sense, JavaGuide offers an alternative ways for students to access the same set of QuizJET questions. A student may choose to go through a regular course portal, select one of the lectures, and choose one of the questions, which a teacher posted to this lecture. Alternatively, a student can go to JavaGuide and select a question with the help of adaptive navigation support. The question presentation and evaluation processes are the same in JavaGuide and "plain".

The interface of JavaGuide consists of the quiz navigation area and the quiz presentation area (Figure 2). Navigation area provides the hyperlinks to QuizJET questions, which are grouped into topics. By clicking on the topic name, a user can expand or collapse questions for the topic. A click on a question loads the question into the quiz presentation area. To assist students in navigating to the appropriate topics and questions the links to each topic are annotated with an adaptive icon.

JavaGuide uses "target-arrow" annotations mechanism of QuizGuide [13], which has been successfully used in a number of C-programming courses [13, 14]. Each adaptively generated target icon expresses two layers of meanings: knowledge adaptation and goal adaptation. For knowledge adaptation, a topic-based modeling is adopted. Each topic contains several educational activities which identified by the course instructor. Student progress with these activities defines the user understanding

of the topic. The number of arrows in the target represents the growth of student knowledge of the topic. Goal adaptation is supported by a time-based mechanism which presents the relevant topics according to course lecture sequence. The color of the target expresses the relevance of the topic to the current course goal. The icon for the current topic is shown as bright blue, its prerequisites are light blue, while target icons for other earlier topics are gray. A crossed icon indicates that the student is not ready for the topic yet.

It is easy to notice that JavaGuide annotation approach integrates prerequisite-based adaptation that advises whether an item is ready or not ready and progress-based annotation that displays the amount of knowledge already acquired by the student. Both approaches are relatively popular and well explored in adaptive educational hypermedia. For example, prerequisite-based approach is used in AHA! [15], ELM-ART [16] and KBS-Hyperbook [17], while progress based annotation is used in INSPIRE [18] and NavEx [19]. An interesting feature of JavaGuide and its predecessor QuizGuide is the use of both annotation approaches in parallel.

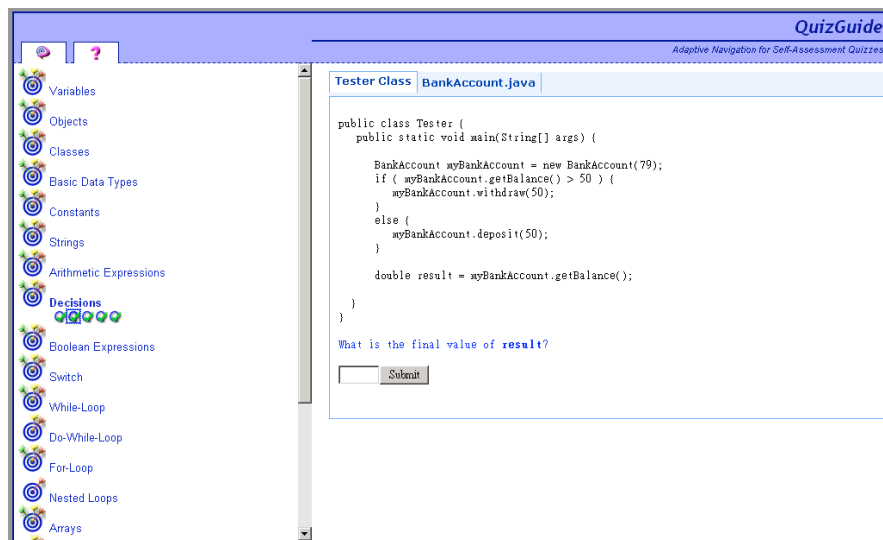


Fig 2. JavaGuide Interface

4 Classroom Studies and Evaluation Results

The classroom study of our technology was performed in two undergraduate introductory programming classes offered by the School of Information Sciences, University of Pittsburgh. Online self-assessment quizzes were used as one of the non-mandatory course tools. QuizJET was used in the Spring semester of 2008 and JavaGuide was used in the Fall semester of 2008. Both tools featured the same set of quizzes. All student activity with the system was recorded. For every student attempt

to answer a question, the system stored a timestamp, the user's name, the question, quiz, and session ids, and the correctness of the answer.

4.1 Basic Statistics

In both classes, student work with the systems was analyzed on two levels: overall and within a session. On each level we explored following performance parameters: Attempts (the total number of questions attempted by the student), Success Rate (the percentage of correctly answered questions) and Course Coverage (the number of attempts by the student for each distinct topic; the number of distinct questions attempted by the student). In addition, we decided to examine all performance parameters separately for students with *weak* and *strong* starting knowledge of the subject. This decision was guided by the results of earlier research [20, 21], which demonstrated that the starting level of knowledge of the subject may affect the impact of adaptive navigation support on student performance. To achieve this separation, the students were split into two groups based on their pre-test scores (ranging from a minimum of 0 to a maximum of 20).

Table 1 compares student performance in QuizJET and JavaGuide. Strong students scored 10 or higher points in the pre-test and weak students scored less than 10 points. The table shows active use of the JavaGuide by both strong and weak students. It also indicates a remarkable increase of all performance parameters in the presence of adaptive navigation support. These results confirm that the impact of adaptive navigation support on student performance, which was originally discovered in the domain of C programming, is sufficiently universal to be observed in a different domain and with a larger variety of question complexity.

Table 1. System Usage Summary

		JavaGuide (Fall 2008)			QuizJET (Spring 2008)
parameters		Strong (n=5)	Weak (n=17)	All Users (n=22)	All Users (n=31)
Overall User Statistics	Attempts	131.2	123.82	125.50	41.71
	Success Rate	58.87%	58.20%	58.31%	32.15%
	Distinct Topics	11.00	12.00	11.77	4.94
	Distinct Questions	41.60	47.53	46.18	17.23
Average User Session Statistics	Attempts	29.82	30.51	30.34	21.50
	Distinct Topics	2.50	2.95	2.85	2.55
	Distinct Questions	9.45	11.71	11.16	8.88

4.2 The Impact of Guidance on Student Work with questions of Different Complexity

Our next goal was to explore the significance of the obtained data and to explore the impact of adaptive navigation support on user work with questions of different

complexity. To do that, all questions were categorized into 3 complexity levels (Easy, Moderate and Hard) based on the number of involved concepts (which in ranged from 4 to as far as 287). A question with 15 or less concepts is considered to be Easy, 16 to 90 as Moderate, and 90 or higher as Hard. In total, both systems included 41 easy, 41 moderate, and 19 hard questions. We conducted two separate 2×3 ANOVA to evaluate student performance measured by *Attempts* and *Success Rate* within two different systems and three complexity levels. The means and standard errors for each group are reported in Table 2.

Table 2. Means and standard error of Attempts and Success Rate, by system and complexity level

DV	Complexity Level	QuizJET	JavaGuide
		(2008Spring) (n=31)	(2008Fall) (n=22)
		<i>M</i> ± <i>SE</i>	<i>M</i> ± <i>SE</i>
Total Attempts	Easy	38.52± 8.40	75.77± 9.98
	Moderate	25.06± 8.40	41.32± 9.98
	Hard	5.58± 8.40	8.41± 9.98
Attempts (per question)	Easy	.94± .22	1.85± .26
	Moderate	.61± .22	1.01± .26
	Hard	.29± .22	0.44± .26
Success Rate	Easy	38.00% ± 5.70%	68.73% ± 6.70%
	Moderate	28.23% ± 5.70%	67.00% ± 6.70%
	Hard	11.90% ± 5.70%	39.32% ± 6.70%

The first 2×3 between-subjects ANOVA was performed on *Attempts* as a function of *System* (QuizJET and JavaGuide) and *Complexity Level* (Easy, Moderate and Hard). We found that JavaGuide received a significantly higher number of *Attempts* than QuizJET, $F(1, 153) = 6.042, p = .015$, partial $\eta^2 = .038$. This result showed that adaptive navigation encourages students to work with parameterized questions. We also found that students had significant higher *Attempts* on the easy quizzes in JavaGuide than in QuizJET, $F(1, 153) = 7.081, p = .009$, partial $\eta^2 = .044$ (Figure 3, left). There were no significant differences found between the systems for the moderate and hard quizzes.

The second set of 2×3 between-subjects analysis of variance was performed on *Success Rate*. We found that with JavaGuide, students achieved significantly higher *Success Rate* than with QuizJET, $F(1, 153) = 40.593, p < .001$, partial $\eta^2 = .210$. The size of effect for the three complexity levels was respectively 1.81 ($p = .001$), 2.37 ($p < .001$) and 3.30 ($p = .002$) times higher than the *Success Rate* in QuizJET. This means that regardless of the complexity of the quizzes, students were, on average, 2.5 times more likely to answer a question correctly when it was accessed with adaptive navigation support than without such support. As shown on Figure 3 right, the *Success Rate* for JavaGuide is dramatically higher than for QuizJET.

The analysis of the impact of adaptive navigation support on student work with questions of different complexity leads to some interesting observation. First, it seems that adaptive guidance encourages students to do more work early in the course when the questions are relatively easy, while also preventing them to venture too fast into the area of very hard questions. Second, the investment of student efforts on work

with easy questions pays back across all three complexity levels. The knowledge gained while working with easy questions helped students to achieve better success in dealing with moderate and hard questions as well. Most pronounced, this effect is in the area of hard questions. While the number of attempts on hard questions is similar in two groups, the success rate with hard questions is more than three times larger in JavaGuide group. Apparently, the prerequisite-based guidance of JavaGuide prepared the students to face complex questions by exploring easier ones.

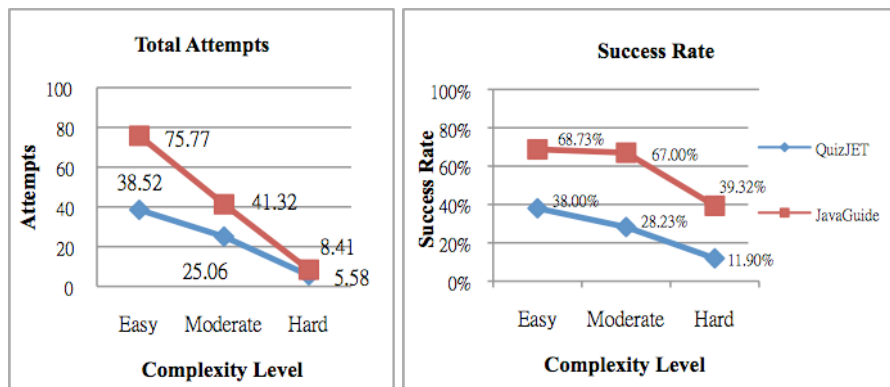


Fig 3. The total *Attempts*(left) and the *Success Rate* (right) of two systems on different complexity levels

4.3 The Impact of Guidance on Weak and Strong Students

As mentioned above, students were categorized as strong students if they have 10 or more points on pre-test and as weak otherwise. We discovered that stronger students had a significantly higher *Success Rate* with the QuizJET system than weaker students did, $F(1, 87) = 4.760, p = .032, \text{partial } \eta^2 = .052$. However, we did not find any significant differences between strong and weak students' *Success Rate* with the JavaGuide system, $F(1, 60) = .007, p = .931, \text{partial } \eta^2 < .001$. With adaptive navigation support, both strong and weak students achieved similar performance on each complexity level of quizzes. Without such support, there was a greater gap between strong and weak students. Thus, adaptive navigation support can, indeed, adapt to the student starting level of knowledge guiding students of both levels to appropriate quizzes. An analysis of the *Attempts* per question uncovers the mechanism behind this observation. The statistics shows that weak students using JavaGuide had a significantly higher number of *Attempts* made in easy questions than they did in QuizJET, $F(1, 147) = 7.658, p = .006, \text{partial } \eta^2 = .050$; while stronger students using JavaGuide had a significantly higher number of *Attempts* in harder quizzes, $F(1, 147) = 4.089, p = .045, \text{partial } \eta^2 = .028$. This suggests that JavaGuide indeed guides students to quizzes that match the students' knowledge: weaker students were more

often guided to work on easy quizzes while stronger students were usually led to work on harder quizzes. Figure 4 shows the pattern of differences found between strong and weak students on various complexity levels for the two systems. The means and standard errors for each group are reported in Table 3.

Table 3. Means and standard error of Attempt per questions and Success Rate by system, complexity level and knowledge level

DV	Knowledge Level	Complexity Level	QuizJET (2008Spring)	JavaGuide (2008Fall)
			M±SE	M±SE
Attempts (per question)	Strong	Easy	1.11 ± .32	1.43 ± .553
		Moderate	.59 ± .32	1.32 ± .553
		Hard	.39 ± .32	.97 ± .553
	Weak	Easy	.78 ± .309	1.97 ± .300
		Moderate	.63 ± .309	.92 ± .300
		Hard	.20 ± .309	.29 ± .300
Success Rate	Strong	Easy	48.27% ± 8.10%	67.80% ± 14.00%
		Moderate	39.40% ± 8.10%	59.80% ± 14.00%
		Hard	12.07% ± 8.10%	49.00% ± 14.00%
	Weak	Easy	28.38% ± 7.80%	69.00% ± 7.60%
		Moderate	17.75% ± 7.80%	69.12% ± 7.60%
		Hard	11.75% ± 7.80%	36.47% ± 7.60%

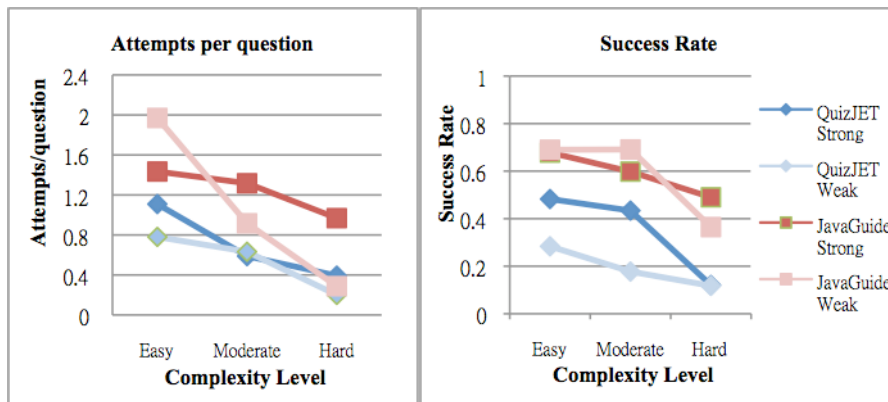


Fig 4. The pattern of differences in the *Attempts per Question* & *Success Rate* for QuizJET and JavaGuide, on a variety of knowledge and complexity levels

4.4 Subjective Evaluation

To evaluate the students' subjective attitudes toward the systems, we administered questionnaires at the end of both semesters. Overall, 97.37% of the students strongly

agreed or agreed that the system should be used again in teaching this course and that the online self-assessment quizzes were relevant to what was presented in class. In terms of learning, 91.12% of the students considered that the self-assessment quizzes contributed to their learning in this course. 71.71% of the students thought that online self-assessment quizzes provided useful feedback. For further improvement on the systems, we also collected students' opinions on system features. 23% of them hoped to have future systems available for handheld computers so that the quizzes can be taken anywhere.

5 Summary and Future Work

This paper explored the impact of adaptive navigation support on student work with parameterized questions for object-oriented programming. The results demonstrated that adaptive navigation encourages students' to use parameterized questions more extensively and significantly increases their success rate. Students were, on average, 2.5 times more likely to answer a question correctly with adaptive navigation support than without such support. Most pronounced was the increase of student work with easy questions. However, encouraging students to work more on easier questions, the navigation support got them much better prepared to face complex questions as well. By categorizing the students' initial knowledge levels into strong and weak, we found that adaptive navigation support effectively guided both strong and weak students to the appropriate quizzes and contributed to a uniformly high success rate.

According to the subjective evaluation, students perceived the online self-assessment quizzes as helpful to their learning. Most of them appreciated the systems. However, about a quarter of the users considered feedback from the self-assessment quizzes to be lacking. These results provoke several new challenges and give us new directions for improvement in the future. Since JavaGuide and QuizGuide were developed from the same set of theories, we are also interested in investigating the differences between online self-assessment quizzes in C and Java. To understand more about the educational values in this context, we plan to perform a more exhaustive evaluation of the systems.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grant No. 0447083.

References

1. Brusilovsky, P., & Sosnovsky, S., Individualized Exercises for Self-Assessment of Programming Knowledge: An Evaluation of QuizPACK. *ACM Journal on Educational Resources in Computing*, 2005. 5(3).

2. Kashy, E., Thoennessen, M., Tsai, Y., Davis, N.E., Wolfe, S.L., Using networked tools to enhance student success rates in large classes, in 27th ASEE/IEEE Frontiers in Education Conference. 1997: Pittsburgh. p. 233-237.
3. Titus, A.P., Martin, L.W., Beichner, R.J., Web-based testing in physics education: Methods and opportunities. *Computers in Physics*, 1998. 12(Mar/Apr): p. 117-123.
4. Merat, F.L., Chung, D. World Wide Web approach to teaching microprocessors. . in *Frontiers in Education Conference*. 1997. Pittsburgh, PA.
5. Graham, C.R., Swafford, M.L., & Brown, D.J. Mallard: A Java Enhanced Learning Environment. in *WebNet'97, World Conference of the WWW, Internet and Intranet*. 1997. Toronto, Canada.
6. Higgins, C., Gray, G., Symeonidis, P., Tsintsifas, A.: Automated assessment and experiences of teaching programming. *ACM Journal on Educational Resources in Computing* 5, 3 (2005) Article No. 5
7. Douce, C., Livingstone, D., Orwell, J.: Automatic test-based assessment of programming: A review *ACM Journal on Educational Resources in Computing* 5, 3 (2005) Article No. 4
8. Ala-Mutka, K.M.: A survey of automatic assessment approaches for programming assignments. *Computer Science Education* 15, 2 (2005) 83-102
9. Lister, R., Adams, E.S., Fitzgerald, S., Fone, W., Hammer, J., Lindholm, M., McCartney, R., Moström, J.E., Sanders, K., Seppälä, O., Simon, B., Thomas, L.: A multi-national study of reading and tracing skills in novice programmers. *ACM SIGCSE bulletin* 36, 4 (2004) 119-150
10. Sosnovsky, S., Shcherbinina, O., & Brusilovsky, P., Web-based parameterized questions as a tool for learning., in *World Conference on E-Learning*.. 2003, AACE. p. 309-316.
11. Brusilovsky, P., Sosnovsky, S., and Yudelson, M. Addictive links: The motivational value of adaptive link annotation in educational hypermedia. in *Proceedings of 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*. 2006. Dublin, Ireland: Springer Verlag.
12. Hsiao, I., Brusilovsky, P., Sosnovsky, S. Web-based Parameterized Questions for Object-Oriented Programming. in *E-Learn 2008*. 2008. Las Vegas, USA: AACE.
13. Brusilovsky, P., Sosnovsky, S., and Shcherbinina, O., QuizGuide: Increasing the Educational Value of Individualized Self-Assessment Quizzes with Adaptive Navigation Support, in *World Conference on ELearning, E-Learn 2004*, Nall, J. and Robson, R. (eds.) Washington, DC, USA., 2004, p. 1806-1813.
14. Brusilovsky, P., Sosnovsky, S., and Yudelson, M. Adaptive Hypermedia Services for E-Learning. in *Proceedings of Workshop on Applying Adaptive Hypermedia Techniques to Service Oriented Environments at the Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*. 2004. Eindhoven, the Netherlands
15. De Bra, P. and L. Calvi (1998). "AHA! An open Adaptive Hypermedia Architecture." *The New Review of Hypermedia and Multimedia* 4: 115-139
16. Weber, G. and P. Brusilovsky (2001). "ELM-ART: An adaptive versatile system for Web-based instruction." *International Journal of Artificial Intelligence in Education* 12(4): 351-384
17. Henze, N. and W. Nejd1 (2001). "Adaptation in open corpus hypermedia." *International Journal of Artificial Intelligence in Education* 12(4): 325-350
18. Grigoriadou, M., K. Papanikolaou, et al. (2001). *INSPIRE: An INtelligent System for Personalized Instruction in a Remote Environment*. Third workshop on Adaptive Hypertext and Hypermedia, Sonthofen, Germany, Technical University Eindhoven
19. Brusilovsky, P., Sosnovsky, S., and Yudelson, M. (2006) Addictive links: The motivational value of adaptive link annotation in educational hypermedia. In: V. Wade, H. Ashman and B. Smyth (eds.) *Proceedings of 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2006)*, Dublin, Ireland, June 21-23, 2006, Springer Verlag, pp. 51-60

20. Specht, M. and A. Kobsa (1999). Interaction of domain expertise and interface design in adaptive educational hypermedia. Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, Toronto and Banff, Canada
21. Brusilovsky, P. (2003). "Adaptive navigation support in educational hypermedia: the role of student knowledge level and the case for meta-adaptation." *British Journal of Educational Technology* 34(4): 487-497