Web-based Parameterized Questions for Object-Oriented Programming

I-Han Hsiao, Peter Brusilovsky, Sergey Sosnovsky
School of Information Sciences
University of Pittsburgh
Pittsburgh PA 15260
United States
{ihh4, peterb, sas15}@pitt.edu

Abstract. Web-based questions for assessment and self-assessment of students' knowledge are important components of modern E-Learning. Parameterized questions allow to enhance the value of Web-based questions while also decreasing the authoring costs. This paper discusses the problems of implementation and evaluation of online parameterized questions for the domain of object-oriented programming. We present QuizJET – a system supporting authoring, delivery, and automatic assessment of parameterized online quizzes and questions—aimed at teaching a non-formula-based domain, Java programming language. The classroom evaluation of QuizJET demonstrated that by working with the system students were able to improve their scores on in-class weekly quizzes. We have also observed a significant relation between the amount of work and the success rate of students and their scores on the final exam.

1. Introduction

Online quizzes formed by questions of different kinds are now the primary tool for evaluation and self-evaluation of student knowledge in the context of Web-based education (Brusilovsky & Miller 2001). Yet traditional Web-based questions have several shortcomings, which are now in the focus of modern e-learning research. One of these directions known as *parameterized* or *individualized* questions, addresses the issue of authoring costs and plagiarism. Unlike a regular question, which is presented exactly as it was authored to all students, a parameterized question is a pattern of a question created by an author. At the presentation time, the pattern is instantiated with randomly generated parameters. As a result, every question pattern is able to produce a large or even unlimited number of different questions. In an assessment context a reasonably small number of question patterns can be used to produce individualized assessments semester by semester even for large classes resolving the potential plagiarism problems, as first shown in CAPA (Kashy et al. 1997). In a self-assessment context, the same question can be used again and again with different parameters allowing every student to achieve mastery. Parameterized questions allowed to achieve these benefits in both context without increasing authoring costs. A number of pioneer systems such as CAPA (Kashy et al. 1997), WebAssign (Titus, Martin & Beichner 1998), EEAP282 (Merat & Chung 1997), or Mallard (Graham, Swafford & Brown 1997) have explored the use of individualized questions in a range of topics and demonstrated benefits of this approach.

In our past work, we attempted to expand the ideas of parameterized questions into the domain of teaching programming languages. This domain is more challenging for the application of this technology than "formula-based" domains explored in the early systems. Using am original runtime answer checking approach, we implemented QuizPACK system, capable to deliver Web-based dynamic individualized exercises for procedural programming language C. Since QuizPACK was first reported at E-Learn 2003 (Sosnovsky, Shcherbinina & Brusilovsky 2003), it was heavily used in many programming classes and emerged into students' favorite learning tool. Multiple classroom evaluations demonstrated the educational value of this QuizPACK (Brusilovsky & Sosnovsky 2005). Despite of its educational effectiveness, the broader impact of QuizPACK appeared to be less than we expected due to countrywide curriculum changes: over the last eight years, the majority of introductory programming classes switched from C to Java programming language. This motivated our further work presented in this paper.

Here we present the results of our more recent work on QuizJET (Java Evaluation Toolkit for Quizzes), a system, which supports authoring, delivery, and evaluation of parameterized quizzes and questions for Java.

QuizJET applies a set of ideas, which we explored in our original work on QuizPACK to the more sophisticated domain of object-oriented Java programming. Due to a number of differences between procedural and object-oriented programming, we have to address several new problems when developing QuizJET. This paper introduces QuizJET system, presents our approach to implementation of parameterized questions for object-oriented programming, and reports the results of the first classroom evaluation of QuizJET.

2. QuizJET: A System to Author, Deliver, and Assess Parameterized Questions for Java

QuizJET (http://adapt2.sis.pitt.edu/quizjet/) was designed as a generic system to author, deliver, and assess a range of parameterized questions for Java programming language. QuizJET can work in both assessment and self-assessment modes and is able to cover a whole spectrum of Java topics from Java language basics, to such critical topics as objects, classes, interfaces, inheritance, and exceptions. At the moment of writing, QuizJET includes 101 question templates grouped into 21 quizzes. Altogether, these questions cover the key topics of the introductory programming class. Each question or quiz can be accessed using a unique URL and is delivered from QuizJET server, which supports both generation and assessment of questions. In our classes, students access the questions through the course portal (Figure 1). Course portal allows a teacher to structure the course as a sequence of lectures or topics and add links to interactive content, relevant to a specific topic, directly to the topic *folder* on the portal. This arrangement allows students working on a specific lecture or topic to access easily all relevant content.

Once a student accesses QuizJET question, the QuizJET server generates an instance of a question and presents it to the student in the frame to the right of the portal's list of topics and activities (Figure 1). QuizJET question presentation screen (Figure 1) includes a small Java program (known also as a *driver class*), a text of question, which typically asks to predict the final value of a specific variable, and the answer input field. It also provides a tabbed interface to access the full code of all classes, which form this Java program including the driver class (which is always shown in first tab) and all imported classes. For example, on Figure 1, a student can access the code of the imported class, *BankAccount*, by clicking on the second tab page. This arrangement is unique for teaching object-oriented programming where even simple object-oriented programs may require one or more imported classes.

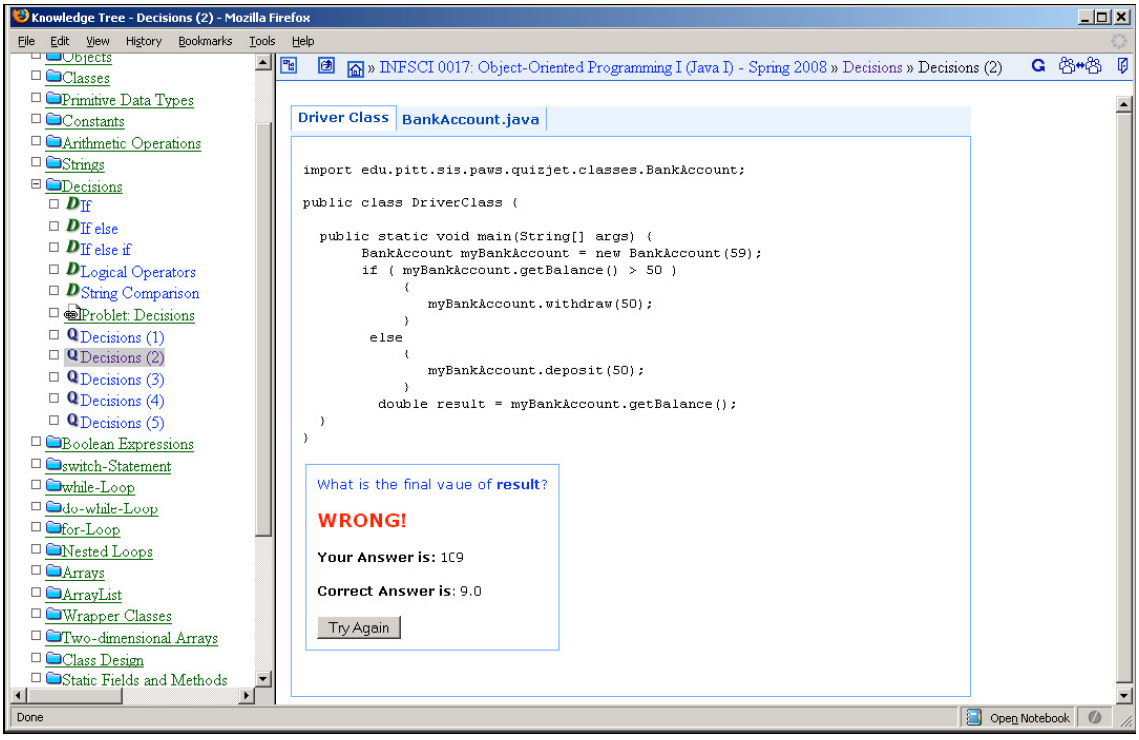


Figure 1: Question presentation in QuizJET

To answer the question, students have to fill in the answer box and submit the answer. The system immediately reports the results of evaluation and presents the correct answer (Figure 2a). Note that the important ability to show correct answer without compromising the question is one of the advantages of the parameterized approach. In this way, the student knows that they are wrong, but are unable to mindlessly memorize the correct answer. Regardless the correctness of the results, users can hit the "Try Again" button and attempt to answer the same question with different parameters. This functionality helps student to eradicate possible misconceptions and to master the course topics.

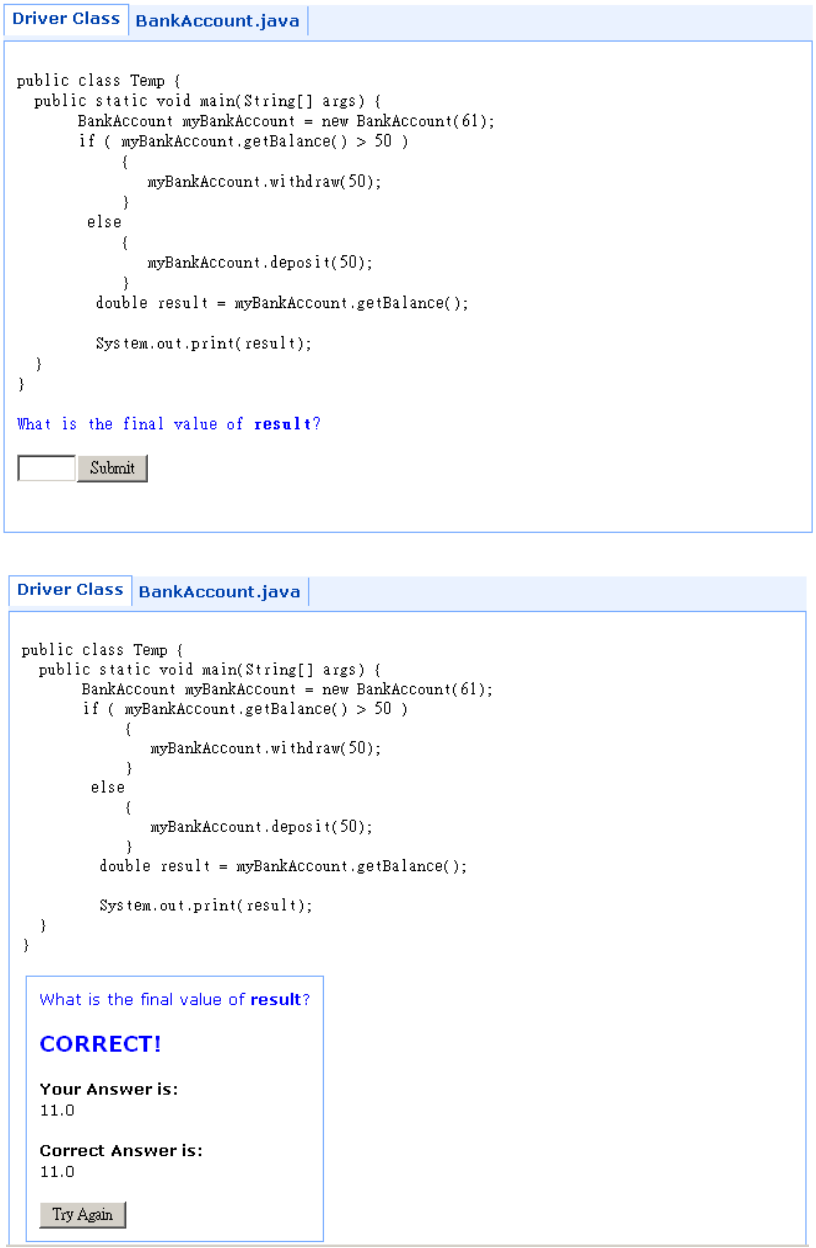


Figure 2: Question interface (top) and question assessment results (bottom) produced by QuizJET

QuizJET question assessment approach follows the ideas of QuizPACK (Brusilovsky & Sosnovsky 2005). The main idea of this approach is to run the parameterized program on the server to determine the correct answer to the generated question. This is important since the answers for different instances of the same question are typically different. This allows both: to assess the correctness of the student answer and to present the correct

answer to the student. In QuizJET this approach become a bit more sophisticated, since the program to run on the server may be composed from several classes, including the driver class with instantiated parameters. Despite several technical differences, the overall presentation and assessment process in both systems look similarly. After a specific question is invoked by its URL, QuizJET randomly generates the question parameter and creates the presentation of the parameterized question (Figure 1). When the student answer is submitted, QuizJET compares student's input to the correct answer which QuizJET produces by running the parameterized driver class code "behind the stage" and presents the results to users on the web (Figure 2). At the same time, it sends the information about the student attempt to the user modeling server.

To produce new quzzes and questions, QuizJET offers a form-based online authoring interface (Figure 3). The current version of the interface allows instructors to create several types of questions such as examining the final value of a variable or predicting the text being printed. Figure 5 shows the question authoring form with a number of fields required to define a question template. An author has to give a *Title* for the question template and specify which *Quiz* it belongs to. The *rdf ID* is an unique attribute for URI to reference to the question template. The body of the question template should be provided in the Code field. In the code, the *_Param* variable indicates where the randomized parameter will be substituted. *Maximum* and *Minimum* specify the interval for the parameter generation. Privacy (set to public by default), indicates the availability of the question to QuizJET users.

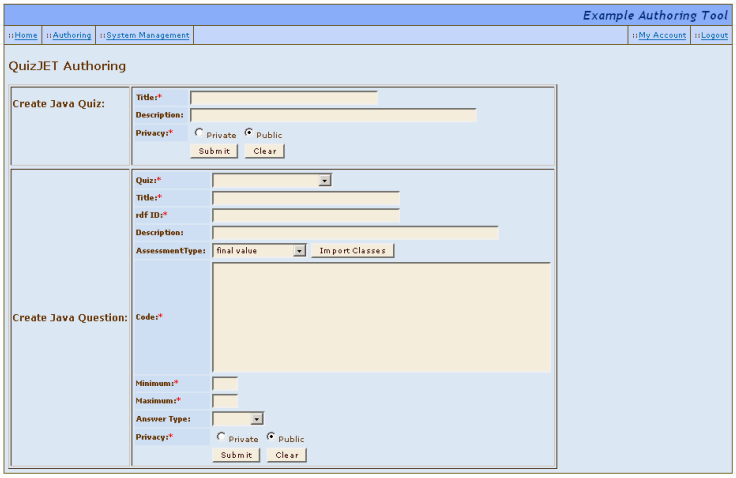


Figure 3: The authoring interface of QuizJET

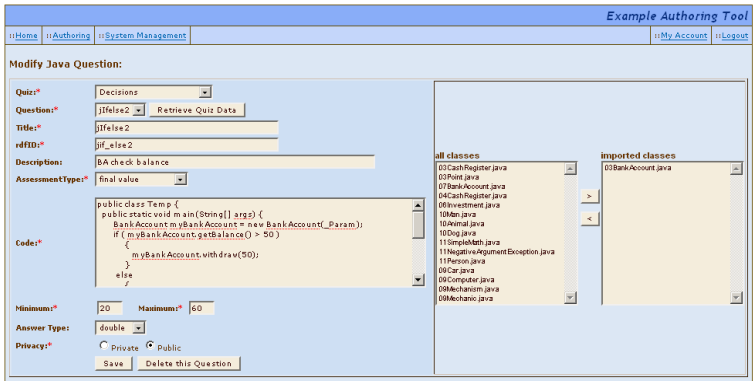


Figure 4: A fully authored QuizJET parameterized question

3. System Evaluation

To explore the value of QuizJET, we performed a classroom study of the system. QuizJET was introduced to the students of undergraduate introductory programming course offered by the School of Information Sciences (University of Pittsburgh) in the Spring semester of 2008. The course focused on the

basics of object-oriented programming with Java language. QuizJET was used as one of the supplementary course tools, i.e., working with QuizJET was non-mandatory. All student activity with QuizJET was recorded over the semester. Every time a student answered a question, the system stored the time of the answer, the user name, the question and the quiz ids, the session id, and the correctness of the answer (correct/incorrect).

3.1 Basic Statistics

Out of 31 students in the course, sixteen chose to work with QuizJET. Some of them tried the system only once, others used it on a regular basis. Eleven of these 16 students used QuizJET quite actively (answered 30 or more questions). In the following analysis, these students are referred to as *active users*.

Table 1 presents a summary of student work with the system. Student performance was analyzed on two granularity levels: overall performance and session performance. On each level we explored two groups of student performance parameters: Student Activity (attempts, success rate) and Course Coverage (topics, questions). The *attempts* variable measures the total number of questions attempted by the student in QuizJET while *success rate* represents the percentage of correctly answered questions calculated as number of correctly answered question divided by the total number of questions attempted. As the table shows, on average system users made 41.71 attempts to answer 17.23 distinct questions. This represents a rather active usage of the system and its parameterized nature. The average success rate was relatively high: 32.15%. The users tried on average 4.94 distinct topics.

	Usage parameters	**All Users(n=31)**	**Active Users(n=11)**
Overall User Statistics	Attempts	41.71	112.00
	Success Rate	32.15%	63.81%
	Distinct Topics	4.94	12.55
	Distinct Questions	17.23	46.00
Average User Session Statistics	Attempts	21.50	24.62
	Distinct Topics	2.55	2.76
	Distinct Questions	8.88	10.11

Table 1: QuizJET usage

3.2 Educational Value of QuizJET

In order to find relationships between students' work with QuizJET and their learning, we examined correlation between student two QuizJET activity parameters (attempts and success) and three parameters of course performance: weekly quiz scores, assignment scores and final exam scores. Quiz, assignment and exam are respectively used as three different dependent variables, all measured in points (for all three variables, maximum number of points were 100; minimum number of point was 0). Since QuizJET was introduced to the class in the middle of term, only the weekly quizzes administered after system introduction were taken into account in the analysis.

	Pearson Correlation	
Variables	**r**	**Sig.**
Quizzes/ Attempts	.284	.122
Quizzes/Success rate	.359	.047*
Assignments/ Attempts	.242	.190
Assignments/Success rate	.176	.343
Final Exam/ Attempts	.445	.036*
Final Exam/Success rate	-.018	.922

Table 2: Correlations between dependent variables and activity, *p<0.05

The correlations between three dependent variables and QuizJET activity and success rate are reported in Table 2. We found significant correlations between taking quizzes and Success rate ($r=0.359$, $p<.05$) and between final exam and attempts ($r=0.445$, $p<.05$). In brief, the more quizzes the students solved, the better successful rate was obtained; the more work with QuizJET is done, the better final exam scores were achieved. The fact that no correlation was found between student performance and assignment scores is not surprising: homework assignments focused on program writing skills, while QuizJET focused on program understanding skills. Interesting, is that the students apparently recognized the value of system as a tool for exam preparation. From the QuizJET usage traffic analysis (Figure 5), we can see that the usage of the system by both 11 active students and the rest of the students marked as "lazy" increased during the last weeks of the course right before the final exam.

To further investigate how users work with QuizJET, we have used SPSS software to perform a Linear Regression Analysis. The dependent variables are quizzes, assignments and final exam. The independent variables are Attempts and Success rate. Unfortunately, we did not find any significance while added Success rate into the regression model.

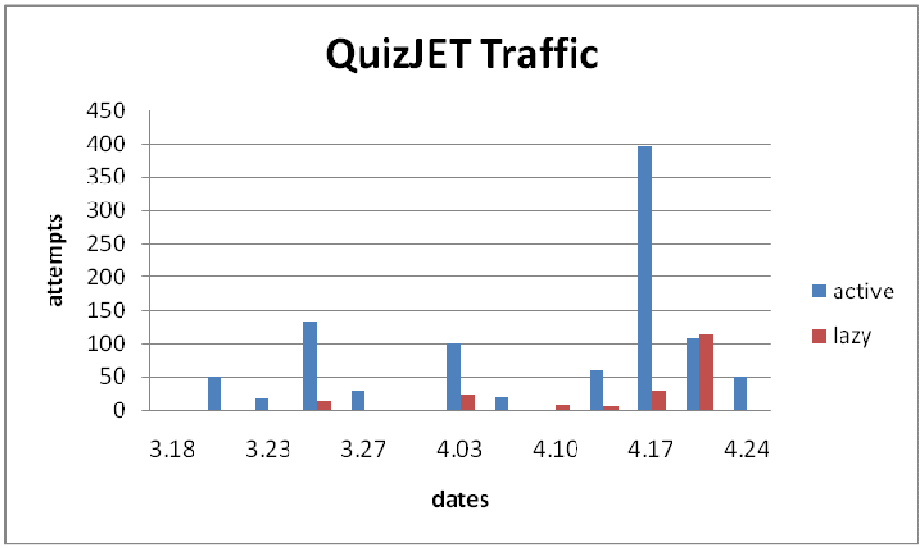
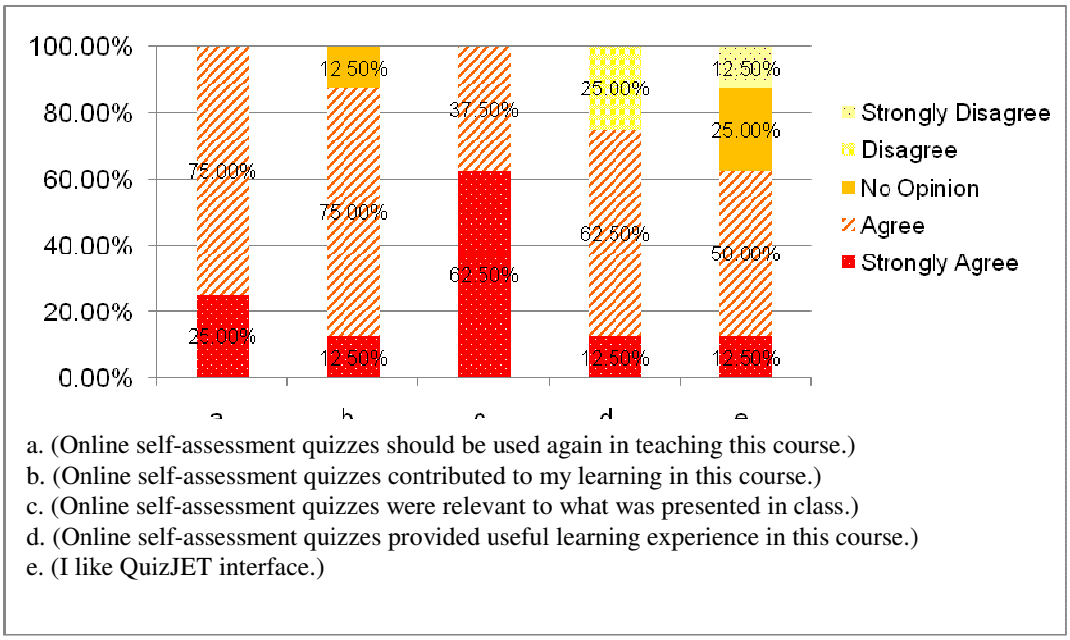


Figure 5: QuizJET usage traffic by class date

3.3 Subjective Evaluation

To obtain student subjective feedback about the system and its features, a non-mandatory questionnaire was administered at the end of the course. Eight out of 16 QuizJET filled in the questionnaire. 68.75% of them were active users. Overall results are illustrated in Figure 7. 100% of the students strongly agree or agree that the on-line assessment quizzes were relevant to what was presented in class and that QuizJET should be used again in teaching this course. 87.5% of the students found that QuizJET was helpful in understanding. The questions presented to them in QuizJET were helpful and contribute to their learning in the class. Typically, a java program is presented by several classes. In our design, due to the reason of limited screen space, we chose to display classes into tab pages. Students found it clear in presentation and were able to master it.



a. (Online self-assessment quizzes should be used again in teaching this course.)
b. (Online self-assessment quizzes contributed to my learning in this course.)
c. (Online self-assessment quizzes were relevant to what was presented in class.)
d. (Online self-assessment quizzes provided useful learning experience in this course.)
e. (I like QuizJET interface.)

Figure 7: Students' attitude to QuizJET

4. A Brief Review of Similar Work

Our work presented in this paper combined two ideas: parameterized question generation and automatic assessment of student answers. The problem of automatic assessment in the area of programming has been explored in the number of projects. A review and classification of this direction of work can be found in (Brusilovsky & Higgins 2005). The majority of projects, however, focused on assessment of programming assignments, which is the most time-consuming activity for instructors and graders. A good review of this work can be found in (Ala-Mutka 2005; Higgins et al. 2005). Automatic assessment of smaller-scale questions and quizzes received less attention, most likely because this need is supported to some extent by traditional quizzes. The style of questions used in QuizPACK and QuisJET: predition of program execusion results, is among the most popular types of questions in programming. The importance of these questions was stressed in the recent SIGCSE working group report (Lister et al. 2004). Despite this known importance, only a handful or projects focused on automatic assessment of this style of questions. In addition to QuizGUIDE, most notable projects include Ramapo college Problets (Dancik & Kumar 2003; Krishna & Kumar 2001; Singhal & Kumar 2000), and Web-To-Test (Arnow & Barshay 1999), which emerged into a commercial product (http://turingscraft.com).

In contrast, parameterized question generation received relatively little attention in the domain of programming. As mentioned in the introduction, the majority of work on question generation has been done in such "formula-based" domain as math or physics. Yet, a number of Ramapo College problets (Krishna & Kumar 2001; Kumar 2005) explore the idea of parameterized generation on a larger scale than QuizPACK and QuizJET and demonstrate the effectiveness of this technology.

5. Summary & Future Work

In this paper, we reported our work web-based parameterized questions for object-oriented programming. We presented the tool, QuizJET, which supports authoring, generation and assessment of online questions for Java programming language. The classroom evaluation of QuizJET uncovered a relationship between QuizJET activity and both weekly quizzes and final exam scores. The QuizJET activity and success of the active user contribute significantly to the regression model to predict final exam score. In addition, subjective survey questions showed that students positively assessed the system and its key features.

We plan to continue our study of QuizJET in classroom context. In our future studies, we expect to collect more representative sampling and be more specific about the way we measure the knowledge gained through the course. We also plan to do the cross comparison between QuizJET and QuizPACK, from course coverage to the structure of complexity of the questions.

Acknowledgements

This work is supported in part by the National Science Foundation under Grant IIS-0447083.

References

Ala-Mutka, K.M. (2005). A survey of automatic assessment approaches for programming assignments. Computer Science Education, 15 (2), 83-102.

Arnow, D., & Barshay, O. (1999). WebToTest: On-line programming examination using WebToTeach. The 4th Annual SICSE/SIGCUE Conference on Innovation and technology in Computer Science Education, ITiCSE'99. SIGCSE Bulletin - Inroads, 21-24.

Brusilovsky, P., & Higgins, C. (2005). Preface to the special issue on automated assessment of programming assignments. ACM Journal on Educational Resources in Computing, 5 (3), Article No. 1.

Brusilovsky, P., & Miller, P. (2001). Course Delivery Systems for the Virtual University. In Tschang, T., & Della Senta, T. (Eds.), Access to Knowledge: New Information Technologies and the Emergence of the Virtual University. Amsterdam: Elsevier Science. 167-206.

Brusilovsky, P., & Sosnovsky, S. (2005). Individualized Exercises for Self-Assessment of Programming Knowledge: An Evaluation of QuizPACK. ACM Journal on Educational Resources in Computing, 5 (3), Article No. 6.

Dancik, G., & Kumar, A.N. (2003). A tutor for counter-controlled loop concepts and its evaluation. 2003 Frontiers in Education Conference (FIE 2003). Session T3C.

Graham, C.R., Swafford, M.L., & Brown, D.J. (1997). Mallard: A Java Enhanced Learning Environment. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 634-636.

Higgins, C. et al. (2005). Automated assessment and experiences of teaching programming. ACM Journal on Educational Resources in Computing, 5 (3), Article No. 5.

Kashy, E. et al. (1997). Using networked tools to enhanse student success rates in large classes. 27th ASEE/IEEE Frontiers in Education Conference, Stipes Publishing L.L.C. 233-237.

Krishna, A., & Kumar, A.N. (2001). A problem generator to learn expression evaluation in CS I and its effectiveness. The Journal of Computing in Small Colleges, 16 (4), 34-43.

Kumar, A. (2005). Rule-Based Adaptive Problem Generation in Programming Tutors and its Evaluation. Workshop on Adaptive Systems for Web-based Education at 12th International Conference on Artificial Intelligence in Education, AI-Ed'2005, IOS Press. 35-43.

Lister, R. et al. (2004). A multi-national study of reading and tracing skills in novice programmers. ACM SIGCSE bulletin, 36 (4), 119-150.

Merat, F.L., & Chung, D. (1997). World Wide Web approach to teaching microprocessors. FIE'97, Frontiers in Education Conference, Stipes Publishing L.L.C. 838-841.

Singhal, N., & Kumar, A.N. (2000). Facilitating Problem-Solving on Nested Selection Statements in C/C++. Frontiers in Education (FiE 2000), IEEE Press.

Sosnovsky, S., Shcherbinina, O., & Brusilovsky, P. (2003). Web-based parameterized questions as a tool for learning. World Conference on E-Learning, E-Learn 2003, AACE. 309-316.

Titus, A.P., Martin, L.W., & Beichner, R.J. (1998). Web-based testing in physics education: Methods and opportunities. Computers in Physics, 12 (Mar/Apr), 117-123.