doi: 10.1111/j.1365-2729.2010.00365.x

Guiding students to the right questions: adaptive navigation support in an E-Learning system for Java programming

I.-H. Hsiao, S. Sosnovsky & P. Brusilovsky

School of Information Sciences, University of Pittsburgh, Pittsburgh, PA, USA

Abstract Rapid growth of the volume of interactive questions available to the students of modern E-Learning courses placed the problem of personalized guidance on the agenda of E-Learning researchers. Without proper guidance, students frequently select too simple or too complicated problems and ended either bored or discouraged. This paper explores a specific personalized guidance technology known as adaptive navigation support. We developed JavaGuide, a system, which guides students to appropriate questions in a Java programming course, and investigated the effect of personalized guidance a three-semester long classroom study. The results of this study confirm the educational and motivational effects of adaptive navigation support.

Keywords adaptive annotation, adaptive navigation support, E-Learning, Java programming, personalized guidance.

Introduction

The developers of modern E-Learning courses strive to offer students more interactive and engaging content, which goes beyond a simple set of static pages. Most frequently they chose to enhance course content with interactive problems of various kinds, from simple questions, to programming exercises (Brusilovsky & Higgins 2005; Douce *et al*. 2005), which could be automatically evaluated by the host E-Learning system. Interactive questions are known to be both engaging and useful in E-Learning context. In a self-assessment mode, they allow the students to check their understanding and discover knowledge gaps. In an assessment mode, they allow instructors to control student learning and certify their progress. All major course management systems provide tools for authoring interactive automatically evaluated questions. In addition, a range specialized authoring and delivery tools allow course creators to include more sophisticated problems and questions. As a result, students taking advanced E-Learning courses have nowadays access to a relatively large number of questions and problems for both assessment and self-assessment.

While the abundance of questions allows students to check various aspects of their learning, this benefit may be not fully realized unless it can guide students to the right questions at the right time as a skilful human tutor does. Without proper guidance, students frequently select too simple or too complicated problems and, as a result, become either bored or discouraged. Unfortunately, 'one-size-fits-all' solutions of this guidance problem (such as, ordering questions in a fixed sequence) do not work since students typically have different starting knowledge and learn at different paces. To remedy this problem, an adaptive guidance should be provided according to the current state of student's knowledge. The two most popular methods of

Accepted: 4 May 2010

Correspondence: I.-H. Hsiao, School of Information Sciences, University of Pittsburgh, Pittsburgh, PA 15260, USA. Email: ihh4@pitt.edu

personalized guidance are adaptive problem generation (Koffman & Perry 1976; Fischer & Steinmetz 2000; Kumar 2005b; Myller 2006) and adaptive problem selection (Mayo & Mitrovich 2000; Mitrovic & Martin 2004; Kumar 2006; Ullrich *et al*. 2009); both of which allow students to focus on problems of optimal difficulty. The negative side of both these approaches is their restrictive nature: they make the adaptive choice for the students leaving them no freedom over the selection process. A potential side effect of such strategy is the student's inability to alter an improper problem selection, which may happen if the student model has been incorrect. In our past work (Brusilovsky & Pesin 1994; Brusilovsky & Sosnovsky 2005b), we explored a less restrictive strategy of adaptive guidance – adaptive navigation support for selecting questions. Adaptive navigation support guides the students to the most appropriate questions by changing the appearance of links to the questions. This approach relies on the synergy between the artificial intelligence (AI) of the system and the students' own intelligence, and often brings better results and higher satisfaction. The evaluation of personalized guidance in self-assessment context (Brusilovsky & Pesin 1994; Brusilovsky & Sosnovsky 2005b) demonstrated that this technology indeed, helps the students to get to the right question at the right time significantly increasing their chance to answer the question correctly. Moreover, we also discovered that the provision of adaptive navigation support dramatically increases the percentage of students actively using educational software, the amount of their work, and frequency of using the system (Brusilovsky *et al*. 2006).

While our past research demonstrated several benefits of using adaptive navigation support for guiding students to the right questions, a number of questions stayed unanswered. First, the quiz questions used in our studies were relatively simple. As a result, it was left unclear whether the benefits of adaptive navigation support are restricted to simple questions or this technology can successfully guide students to a broader range of questions: from relatively simple to very difficult. Second, due to a relatively small number of subjects in our classroom studies, we were not able to separately assess the impact of adaptive navigation support technology on stronger and weaker students, which is a typical research question in the area of AI in Education (Mitrovic 2007). It is known that some educational innovations may be especially beneficial for stronger or weaker students, while others provide equal support to both groups, but earlier research has not explored this aspect of adaptive navigation support technology.

The work presented in this paper attempted to investigate adaptive navigation support for self-assessment questions beyond the original narrow scope, i.e. in larger classes and with a broader range of question difficulty. To allow this expansion, we moved our studies to a new and more sophisticated domain of Java programming language, which is now the language of choice in most introductory programming classes. To form the basis for our study, we developed QuizJET (Java Evaluation Toolkit), a system for authoring, delivery, and evaluation of parameterized questions for Java (Hsiao *et al*. 2008). A preliminary evaluation has demonstrated that QuizJETs' questions are educationally beneficial: we found a significant relationship between the quality and the amount of work done by students in QuizJET and their performance. By using the system, students were able to improve their in-class weekly quiz scores. We also found that their success in QuizJET (percentage of correct answers) positively correlates with scores on the final exam.

Once the effect of QuizJET questions was confirmed, we developed JavaGuide system, which uses adaptive navigation support to guides students to most appropriate QuizJET questions. The effect of adaptive navigation support was evaluated in a three-semester long classroom study, which 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.

The rest of the paper presents our account of this work. After a brief overview of related work, we present the details of both QuizJET's and JavaGuide's implementation, explain the nature of adaptive navigation support, and report the results of classroom studies. We conclude with the summary of results and a brief discussion.

Related work

Parameterized questions in E-Learning

Parameterized questions and exercises emerged as an active research area in the field of E-Learning

(Brusilovsky & Sosnovsky 2005a). This technology allows obtaining 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 (Kashy *et al*. 1997), WebAssign (Titus *et al.* 1998), EEAP282 (Merat & Chung 1997), Mallard (Graham *et al*. 1997) and COMBA (Sitthisak *et al*. 2008), 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. While parameterized questions were mostly used in 'formula-based' problems, we can name a few projects that applied parameterized question generation in E-Learning systems for programming domain (Krishna & Kumar 2001; Kumar, 2000, 2005a; Koffman & Perry 1976; Martin & Mitrovic 2002). In the context of other work on parameterized question generation, our approach could be considered relatively simple and straightforward. Our goal was not to improve problem generation, but to implement a practical and robust solution that can dramatically reduce authoring effort required to create a sizeable collection of questions for teaching programming.

Adaptive navigation support in E-Learning

Adaptive navigation support is a group of techniques that aim to help individual users locate, relevant information in the context of hypertext and hypermedia (Brusilovsky 2001). By adaptively altering the appearance of links on every browsed page, such methods as direct guidance, adaptive ordering, adaptive link hiding and removal, and adaptive link annotation support browsing-based personalized access to information. E-Learning, with its constant need to adapt to the level of student knowledge, is one of the most active application areas of adaptive navigation support. In E-Learning context, these techniques demonstrated their ability to support faster achievement of the users' goals, reduce navigational overhead, and increase user satisfaction (Brusilovsky & Eklund 1998; Davidovic *et al*. 2003; Olston & Chi 2003; Kavcic 2004). However, the majority of systems applying these techniques in E-Learning, as well as the majority of evaluation studies, focused only on guiding students to the right piece of text-based content – such as concept introduction or explanation. In this context, neither the complexity of the content, nor the student's learning success can be measured reliably. In contrast, our work presents one of the very few examples of applying adaptive navigation support to guide students to the most appropriate questions and problems. We believe that such context offers a chance to increase the impact of adaptive navigation support and allows better evaluation of this impact.

QuizJET: parameterized questions for Java

QuizJET system has been designed to support web-based authoring, delivery and evaluation of parameterized questions for Java programming language. QuizJET can be used for assessment and self-assessment of students' knowledge of a broad range of Java topics from language basics to advanced concepts, such as polymorphism, inheritance and exceptions.

In a taxonomy of task types in computing, questions generated by QuizJET belong to the group of prediction tasks (Bower 2008). These tasks are becoming increasingly popular in various computing-related courses (Malmi *et al*. 2005; Kumar 2005a; Myller 2006). To a large extent, the nature of tasks generated by QuizJET follows the approach explored earlier in QuizPACK (Brusilovsky & Sosnovsky 2005a). However, the switch of the domain from C to Java allowed QuizJET to generate questions of much larger complexity, which was essential for our study.

Table 1 presents the comparison of QuizPACK and QuizJET sets of questions developed to cover the introductory programming courses on C and Java correspondingly. The question complexity is measured by the number of concepts involved in the question. For C, this number ranges from 1 to 19; for Java, it is between 4 and 297. As the table shows, the complexity range for C programming questions is relatively small with most

Table 1. Programming language C & Java question complexity.

Complexity level	Language	C	Java
	# of concepts		
Easy	1~15	161	41
Moderate1	16~40	19	20
Moderate2	41~90	0	21
Complex	91~287	0	19

questions falling to the easy group. On the contrary, Java covers a wider spectrum of complexity with a wider question distribution among levels.

QuizJET student interface

A typical QuizJET question consists of a small Java program. One (or several) numeric value in the text of the program is instantiated with a random parameter when the question is delivered to a student. As a result, students can access the same question multiple times with different values of the parameter and different correct answers. To answer a question, students need to examine the program code and solve a follow-up task. The task can take one of two forms: 'What will be the final value of an indicated variable?' or 'What will be printed by the program to the standard output?'

A tabbed interface design has been implemented to allow questions consist of several classes. The driver class, containing the main function, is always presented on the first is the entry point to the question. The first tab also includes the question task and the field for student's input. The system's feedback is also presented in the first tab after a student's answer has been evaluated. A QuizJET question example is presented in Fig 1. By clicking on different tabs students can switch between the classes to access the full code of the program.

Once a student enters an answer and clicks the '*Submit*' button, QuizJET reports the evaluation results and the correct answer (Fig 2). Whether the results were correct or not, the student can click the '*Try Again*' button to assess the same question with a different value of the generated parameters. This option provides students with an opportunity to master a particular topic.

QuizJET architecture

QuizJET has been developed as a component of ADAPT[2] architecture for distributed adaptation and user modeling.[1] It complies with the ADAPT[2] protocols for user authentication, reporting user interaction and adaptation. URLs of QuizJET questions can be augmented with ADAPT[2] HTTP parameters to notify the system about the current user, group and session. Upon verifying student answers, QuizJET also generates a learning event transaction, which contains information about the user, the question, the result of the interaction, etc. The transaction is sent to the user modelling server CUMULATE that computes student knowledge and reports it to the interested systems (Brusilovsky *et al*. 2005). This architecture enables easy integration of QuizJET with value-added adaptation services.

Each QuizJET question is accessible by a unique URL. Once a question is launched, QuizJET server generates a question and delivers it to a student's browser. When the student submits a solution, QuizJET executes the question code to produce the right answer, compares it with the user's input and presents a feedback.

QuizJET question authoring

QuizJET offers a form-based online authoring interface for developing new quizzes and questions. Figure 3 demonstrates the process of QuizJET question

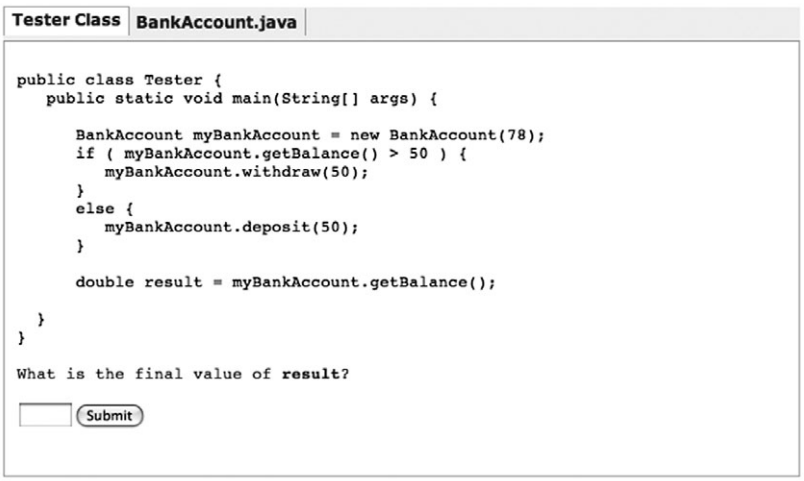

Fig 1 The presentation of a QuizJET question.

Fig 2 The evaluation results of a QuizJET question.

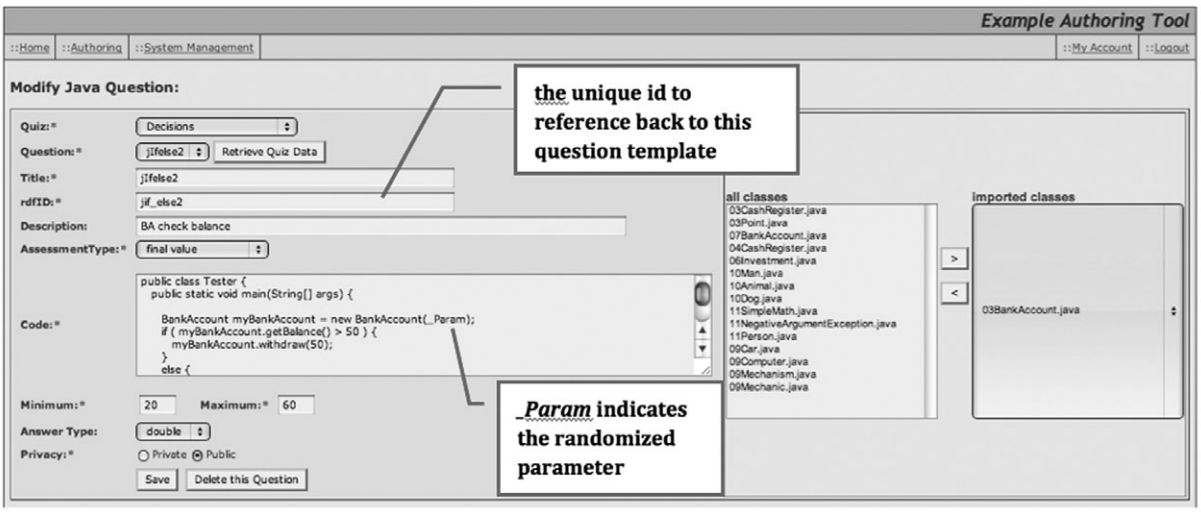

Fig 3 A fully authored QuizJET parameterized question.

authoring. The question template form requires an author to specify several question parameters. An author has to provide the *Title* for the question template and specify which *Quiz* it belongs to. The *rdfID* is a unique attribute to reference the question template. A short comment about the question template can be given under the *Description* field. The *Assessment Type* dropdown box is the attribute, which specifies the task of the question. Currently, there are two forms of the task available: evaluation of the final value of a variable and prediction of what will be printed to the standard output. 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. *Answer Type* dropdown box

provides a list of data types for the final value. *Privacy* indicates the availability of the question to QuizJET users. Currently, QuizJET includes 101 question templates grouped into 21 quizzes. Authors are allowed to upload supplemental classes to include in their questions. Every supplemental class is reusable and is listed on the right hand side of the authoring interface (Fig 3).

JavaGuide: adaptive navigation support for QuizJET questions

The development of QuizJET along with its authoring system, allowed us to create a sufficient volume of questions, which was vital for further experiments with personalized guidance. Our next step was to develop JavaGuide, the system that provides students with personalized guidance to QuizJET questions. The questions in JavaGuide are combined under large topics (from three to six questions per topic) that organize the course material into instructionally complete chunks. Students can browse the material by clicking on topic and question links (Fig 4). A click on a topic link folds/unfolds questions available for the topic. This allows students to organize their learning space more flexibly. A click on a question link loads the corresponding question in the question frame of the system's interface. On both levels – topics and questions – the system offers personalized guidance using adaptive link annotation, one of the most popular adaptive navigation support techniques.

On the topic level, JavaGuide uses a specific form of adaptive link annotation inspired by the ideas of open learner modelling: it presents to a student the content of her/his user model in the form of navigational cues. Every topic link annotation represents the current state of a student's knowledge for the topic. As a result, a student is constantly aware of his/her performance and is able to focus on those parts of the course in which he/she has not demonstrated enough progress.

Topic-level adaptive annotations are visible to students as 'target-arrow' icons (Fig 5). The icons deliver two kinds of information to the student: the individual performance of the student with the topic's content and the relevance of the topic to the current learning goal of

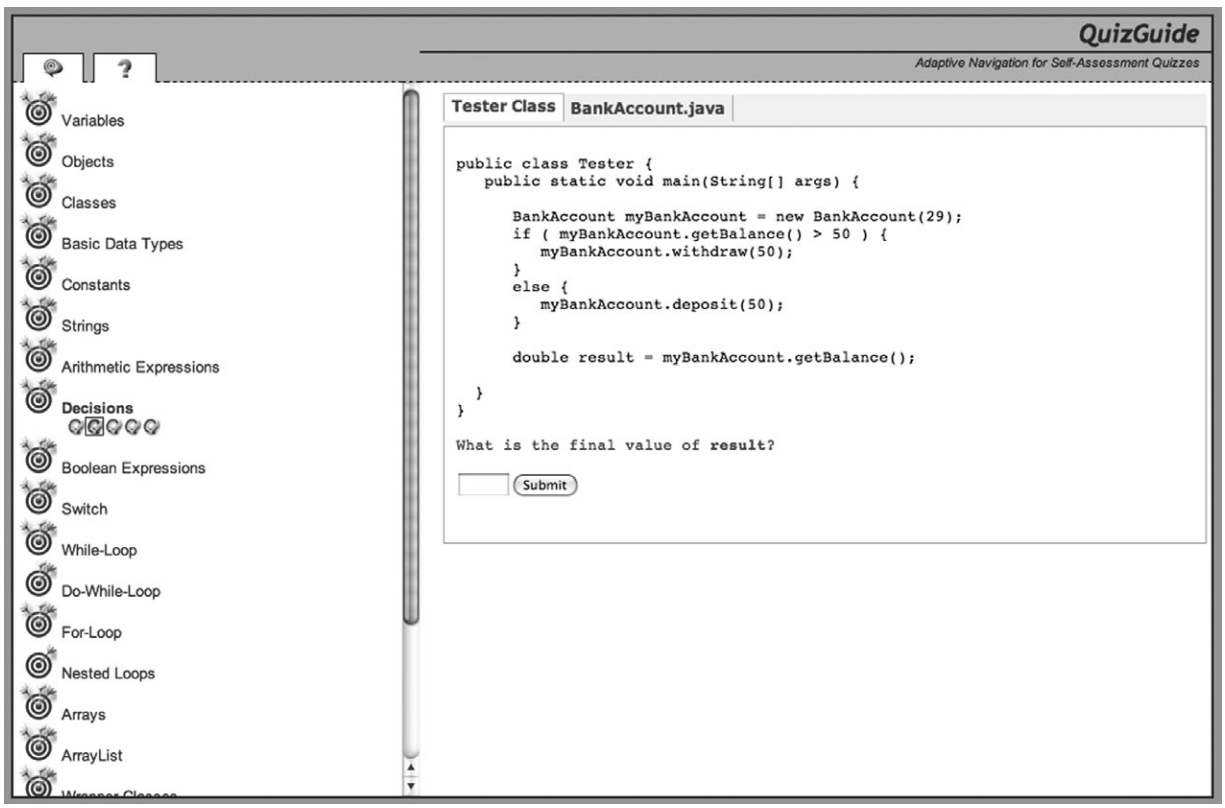

Fig 4 JavaGuide interface.

Fig 5 Upper row: the level of relevance to the current learning goal (current goal, prerequisite for the current goal, passed goal, future goal); lower row: levels of knowledge for the topic.

the entire course. The number of arrows (from 0 to 3) in the target reflects the progress demonstrated for the topic. Once the student has solved enough questions correctly, the topic will be annotated with the '3-arrows target', which indicates the highest level of mastery and tells the student that he/she should focus on a different topic. If no or very little progress has been made on the topic, the target icon for this topic will be empty, which invites the student to concentrate on this topic more.

The colour of the topic icon designates the relevance of the topic to the current learning goal (Fig 5). As new topics are introduced by the teacher of the course, JavaGuide annotates them with bright-blue icons representing the current learning goal of the students. Topics that have been introduced earlier in the course are no longer relevant to the current goal. JavaGuide indicates so by annotating them with grey icons. If a student has problems with any of the past topics that need to be mastered in order to understand the current learning goal, he/she most probably will have problems with the current topics as well. To support students in resolving such problems, JavaGuide annotates topics that are prerequisites for any of the current learning goals with pale-blue target icons. Finally, all the topics that have not been introduced in the course yet are annotated with crossed-out target icons; this means the student is not ready for them yet.

Thus, the topic annotations in JavaGuide combine two kinds of adaptation: individual progress-based adaptation and group-wise time-based adaptation. JavaGuide does not restrict the access to the learning content in any way. The students can access any topics, even those that have not been introduced yet. JavaGuide merely informs the students about the individual and group-wise importance of the topics and tries to direct students to the best learning content at any particular moment of time.

To help the student understand the meaning of all elements of the interface, JavaGuide dynamically generates mouse-over hints for the icons. A detailed help explaining all interface elements is available as well.

To further assist students in navigating through the corpus of available learning content, JavaGuide also supports adaptive annotation for individual questions. Question icons of JavaGuide report to students the completion status of questions. The completion status of a question is a binary entity. It reflects whether the specific question has been solved correctly at least once. As soon as a student submits his/her first correct answer to a question, the corresponding icon receives a checkmark. This can help students to choose between similar questions characterized within a topic. If one of the questions has a checkmark, and another does not, a student who is still interested in testing her/his knowledge of this topic will be guided to the unsolved question.

Classroom studies and evaluation results

Experiment participants and evaluation method

In order to explore the value of adaptive navigation support in the context of Java programming, we performed three classroom studies. All of them were performed with undergraduate students of the same introductory programming course offered by the School of Information Sciences (University of Pittsburgh). The course focuses on the basics of object-oriented programming with Java language. In the context of this course, QuizJET self-assessment quizzes were used as one of the supplementary course tools. QuizJET without JavaGuide (in non-adaptive mode) was evaluated in the Spring semester of 2008 and with JavaGuide (in adaptive mode) was evaluated in the Fall semester of 2008 and again in the Spring semester of 2009. All three semesters used the same set of quizzes. All student activity with the system was recorded over the semester. Every time a student answered a question, the system stored the timestamp, the user's name, the question, quiz and session ids, and the correctness of the answer.

Table 2 summarizes the descriptive parameters of the student population participating in the studies. Every course had between 30 and 40 students. Female students

Table 2. Study participants.

Semester System	Spring 2008 Non-adaptive	Fall 2008 Adaptive	Spring 2009 Adaptive
Pre-quiz : Post-quiz : Questionnaire	Yes : Yes : Yes	Yes : No : Yes	Yes : Yes : Yes
Number of students:			
Overall	31	38	34
Working with the system	16 (52%)	22 (58%)	19 (56%)
Male/Female student distribution:			
Overall	25/6	27/11	23/11
Working with the system	13/3	16/6	12/7
Weak/Strong student distribution:			
Overall	16/15	30/8	28/6
Working with the system	6/9[1]	14/5[2]	17/2
Average score in the pre-quiz:			
Overall	10.18	4.97	3.19
Working with the system	10.20	5.16	2.68

[1]One of the students who worked with the system in the Spring 2008 semester did not take the pre-test.
[2]Three students working with the system in the Fall 2008 semester did not take the pre-test.

represented about 25–30% of the population, which is usual for programming courses in our school.

Somewhat more than a half of the students worked with the system every semester. The usage of the system was purely voluntary. Students were presented the system in the beginning of the semesters and told that it can help them to learn Java and prepare for in-class quizzes. However, no incentive was administered, and neither the amount nor the character of students' work with the system influenced their grades.

In the beginning of the semesters, students took a pre-quiz evaluating their initial knowledge of Java programming concepts covered by QuizJET questions. The pre-quiz did not change over the semesters. A post-quiz was also administered at the end of Spring 2008 and Spring 2009 semesters to measure students' knowledge gains. The difference between the pre-quiz and the post-quiz was in the numeric values within the questions and the final answers. The structure and the set of the questions did not change. At the end of every semester, we also collected questionnaires that asked students to report their opinion about different features of the system.

System usage parameters

In both classes, students' work with the systems was analysed on two levels: overall and within a session. On each level we explored following system usage 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 distinct topics attempted by the student; the number of distinct questions attempted by the student).

Table 3 compares student performance in three target semesters. The table shows active use of the JavaGuide by the students. It also indicates a remarkable increase of all the system usage parameters in the presence of adaptive navigation support. We found that JavaGuide [$M = 137.17$, standard error (SE) $= 14.85$) received a significantly higher number of *Attempts* than QuizJET ($M = 80.81$, $SE = 23.88$), $F(1, 57) = 4.040$, $P = 0.04$, partial $\eta^2 = 0.068$. This result showed that adaptive navigation encourages students to work with parameterized questions. Hence, the system usage 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.

Relation between working with the system and in-class performance

We have found that students have improved their in-class weekly quizzes scores by working with QuizJET. There is a significant relationship between the amount of work done with the system and the in-class quiz marks. Higher values of *Attempts* correlate

Table 3. System usage summary.

	Parameters	QuizJET (2008 Spring) (n = 16)	JavaGuide (2008 Fall) (n = 22)	JavaGuide (2009 Spring) (n = 19)
Overall user statistics	Attempts	80.81	125.50	144.0
	Success rate	42.625%	58.31%	66.88%
	Distinct topics	9.56	11.77	15.00
	Distinct questions	33.37	46.18	58.42
Average user session statistics	Attempts	21.55	30.34	31.35
	Distinct topics	2.31	2.85	2.55
	Distinct questions	8.9	11.16	8.88

positively with the higher in-class quiz scores (r = 0.359, $P = 0.047$). Higher *Success Rate* also correlates with high scores on the final exam (r = 0.445, $P = 0.036$). These results indicate the educational utility of the work with QuizJET self-assessment quizzes and provide an extra argument in favour of the motivational effect of adaptive annotation reported in the previous subsection. As the amount of work with the quizzes positively correlate with students' in-class performance and adaptive annotations encourage students to do more work, it means that adaptive annotations provided by JavaGuide for QuizJET quizzes positively influences students' learning.

The impact of guidance on student work with questions of different complexity

As we mentioned in the beginning of Section 3, Java domain covers a wider range of question complexity compared with C (see Table 1). Essentially, object-oriented programming is a more complex subject than a procedural language. Thus, it leads to the next research question: '*How do student work with different complexity of questions and how does adaptive navigation support help them*'?

To explore the impact of adaptive navigation support on students' work with questions of different complexity, we have divided all QuizJET questions into three categories (*Easy*, *Moderate* and *Complex* based on the number of involved concepts (that ranged from 4 to 287). A question with 15 or less concepts is considered to be *Easy*, 16 to 90 as *Moderate* and 90 or higher as *Complex (Table 1*). Overall, the developed set of questions includes 41 easy, 41 moderate and 19 hard questions. In order to compare how the two systems helped students to learn with questions of different complexity, we conducted two separate 2×3 analysis of variance (ANOVA). To evaluate their performance we used the familiar parameters *Attempts* and *Success Rate* within adaptive and non-adaptive versions of the systems and complexity levels. The values for means and *SEs* of each group are reported in Table 4.

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 Complex). We found that students had significantly higher *Attempts* on the easy and moderate quizzes in JavaGuide than in QuizJET, $F(1, 162) = 11.498$, $P = 0.001$, partial $\eta^2 = 0.066$; $F(1, 162) = 5.750$, $P = 0.018$, partial $\eta^2 = 0.034$ (Fig 6). There were no significant differences between adaptive semesters (2008 Fall and 2009 Spring), $F(1, 162) = 0.893$, $P = 0.345$, partial $\eta^2 = 0.005$. It demonstrated that adaptive navigation support provides the stable effect to promote attempts across different complexity levels.

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, $F(1, 162) = 72.088$, $P < 0.001$, partial $\eta^2 = 0.308$. The size of effect for the three complexity levels was, respectively, 1.85 ($P < 0.001$), 2.31 ($P < 0.001$) and 3.66 ($P < 0.001$) times higher than the Success Rate in QuizJET. This means that regardless of the complexity of the quizzes, students were, on average, 2.61 times more likely to answer a question correctly when it was accessed with adaptive navigation support than without such support. As shown on Fig 7, the Success Rate for JavaGuide is dramatically higher than for QuizJET. In addition, there were no significances between semesters which used

Table 4. Means and SE of attempts and success rate, by system and complexity level.

DV	Complexity level	JavaGuide (2009 Spring) (n = 19) $M \pm SE$	JavaGuide (2008 Fall) (n = 22) $M \pm SE$	QuizJET (2008 Spring) (n = 16) $M \pm SE$
Total attempts	Easy	73.85 ± 8.33	75.77 ± 9.98	38.50 ± 9.07
	Moderate	60.16 ± 8.33	41.32 ± 9.98	25.06 ± 9.07
	Complex	10.11 ± 8.33	8.41 ± 9.98	5.56 ± 9.07
Attempts (per question)	Easy	1.80 ± 0.21	1.85 ± 0.26	0.94 ± 0.24
	Moderate	1.47 ± 0.21	1.01 ± 0.26	0.61 ± 0.24
	Complex	0.53 ± 0.21	0.44 ± 0.26	0.29 ± 0.24
Success rate	Easy	72.40% ± 5.40%	68.73% ± 6.70%	38.00% ± 5.80%
	Moderate	63.30% ± 5.40%	67.00% ± 6.70%	28.20% ± 5.80%
	Complex	47.80% ± 5.40%	39.32% ± 6.70%	11.90% ± 5.80%

DV = dependent variables; SE = standard error.

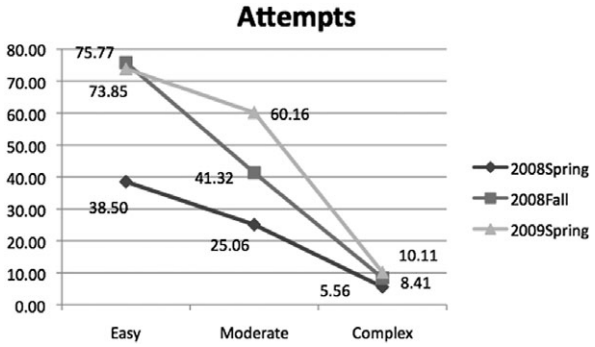

Fig 6 The total *Attempts* of two systems on different complexity levels; 2008 Spring represents QuizJET; 2008 Fall & 2009 Spring represent JavaGuide.

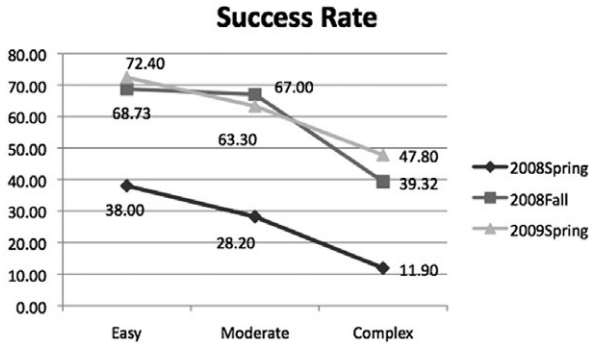

Fig 7 The *Success Rate* of two systems on different complexity levels; 2008 Spring represents QuizJET; 2008 Fall & 2009 Spring represent JavaGuide.

adaptive systems $F(1, 162) = 0.444$, $P = 0.506$, partial $\eta^2 = 0.003$. As shown in Fig 7, the lines of adaptive semesters mingle together. It again proves the adaptive navigation support provided stable guidance to improve students' success rate, not due to the differences in students' distribution.

The analysis of the impact of adaptive navigation support on student work with questions of different complexity leads to some interesting observations. 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 students' efforts into 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. This effect is most pronounced in the area of hard questions. While the number of attempts of the hard questions is similar in two groups, the success rate for the hard questions is more than three times higher in the JavaGuide group. Apparently, the prerequisite-based guidance of JavaGuide prepared the students to face complex questions by exploring easier ones.

The impact of guidance on weak and strong students

The students were split into two groups based on their pre-test scores (ranging from a minimum of 0 to a maximum of 20). Strong students scored 10 or higher points in the pre-test, and weak students scored less than 10 points (see Table 2). We discovered that stronger students had a significantly higher *Success Rate* on easy level questions with the QuizJET system than weaker students, $F(1, 90) = 4.123$, $P = 0.045$, partial $\eta^2 = 0.044$. However, we did not find any significant

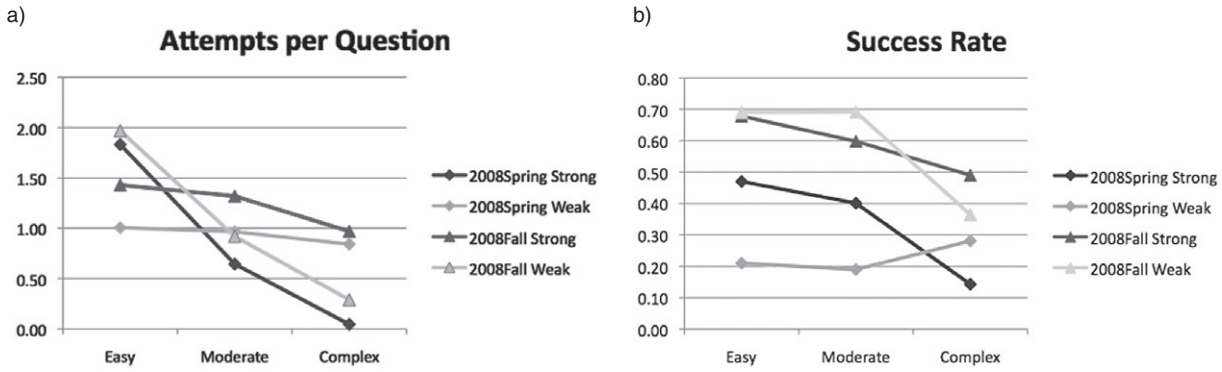

Fig 8 The pattern of differences in the *Attempts per Question* & *Success Rate* for QuizJET and JavaGuide, on a variety of knowledge and complexity levels.

Table 5. Means and standard error of attempt per questions and success rate by system, complexity level and knowledge level.

DV	Knowledge level	Complexity level	QuizJET (2008 Spring) $M \pm SD$	JavaGuide (2008 Fall) $M \pm SD$
Attempts (per question)	Strong	Easy	1.83 ± 0.40	1.43 ± 0.53
		Moderate	0.66 ± 0.40	1.32 ± 0.53
		Complex	0.46 ± 0.40	0.97 ± 0.53
	Weak	Easy	1.00 ± 0.38	1.97 ± 0.28
		Moderate	0.97 ± 0.38	0.92 ± 0.28
		Complex	0.84 ± 0.38	0.29 ± 0.28
Success rate	Strong	Easy	47.00% ± 9.80%	67.80% ± 13.00%
		Moderate	40.00% ± 9.80%	59.80% ± 13.00%
		Complex	14.07% ± 9.80%	49.00% ± 13.00%
	Weak	Easy	21.30% ± 9.20%	69.00% ± 6.70%
		Moderate	19.05% ± 9.20%	69.12% ± 6.70%
		Complex	28.03% ± 9.20%	36.47% ± 6.70%

DV = dependent variables; SD = standard deviation.

differences between strong and weak students' *Success Rate* with the JavaGuide system, $F(1, 90) = 3.305$, $P = 0.072$, partial $\eta^2 = 0.035$. With adaptive navigation support, both strong and weak students achieved similar performance on all levels of question complexity. 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, $F(1, 90) = 4.857$, $P = 0.030$, partial $\eta^2 = 0.051$; while stronger students using JavaGuide had a significantly higher number of *Attempts* in harder questions, $F(1, 90) = 4.147$, $P = 0.045$, partial $\eta^2 = 0.044$. This suggests that JavaGuide indeed guided students to the learning material that matched their knowledge: weaker students were more often guided to work on easy quizzes, while stronger students were usually led to work on harder quizzes. Figure 8a and b shows the pattern of differences found between strong and weak students on various complexity levels for the two systems. The means and standard deviations for each group are reported in Table 5. It also contrasts the performance of weak and strong students in the presence of adaptive navigation support.

Subjective evaluation

To examine the students' attitudes toward the systems, we collected questionnaires at the end of each semester. The responses from students who actually used the system over a semester have been analysed. Eight students filled-in the questionnaire in the Spring semester of 2008 (non-adaptive quizzes served by QuizJET). Twenty-three students (11 in the Fall semester of 2008 and 12 in the Spring semester of 2009) answered to the JavaGuide questionnaire. Overall, six respondents were strong students; 12 were female students. No significant difference was found between the answers of strong and week students, neither between the answers of male and female students.

The questionnaires consisted of different sets of questions across the semesters. Figure 9 presents summary of the questions that stayed in all semesters' questionnaires. No significant difference was observed between the answers of students in different semesters. Overall, students' attitude towards the system was very positive. Of the students, 97.77% agreed or strongly agreed that QuizJET's self-assessment quizzes should be used in this course in the future. Of the system users, 87.10% believed that it helped them better understand difficult concepts of the course. This is an important observation for a knowledge-intensive subject such as Java programming. Students valued QuizJET quizzes as a useful extracurricular learning tool; 91.30% of them reported that the system helped them to prepare for exams. The same 91.30% also believed that QuizJET quizzes contributed to their learning in the course. Such a positive evaluation received from the real users of the system is very encouraging; especially considering that it has been stable over the three semesters and has come from various categories of students (strong/week, female/male).

Summary and future work

This paper investigated the use of adaptive navigation support for guiding students to most appropriate self-assessment questions. Expanding our earlier work, we explored the value of adaptive navigation support in a new domain of much larger complexity: object-oriented Java programming. We have found that adaptive navigation support helps to promote students' participation and significantly increases their success rate with online self-assessment quizzes. Students were, on average, 2.61 times more likely to answer a question correctly with adaptive navigation support than without it. We also found that adaptive navigation support effectively guided both strong and weak students to the appropriate quizzes and contributed to students to more difficult quizzes.

According to the subjective evaluation, students perceived the online self-assessment quizzes as helpful to their learning. Most of them appreciated the systems. Student answers also pointed to the most important directions of improving the system. In particular, about a quarter of the users indicated that a better feedback for the self-assessment questions should be provided. Some students also suggested making the systems available for mobile devices. These results provoke several new challenges and give us guidance for future work.

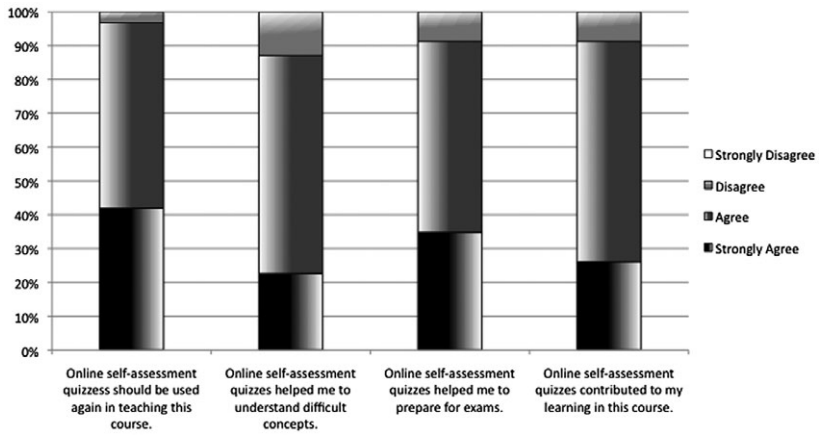

Fig 9 Subjective evaluation of QuizJET quizzes.

Acknowledgement

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

Note

[1]Description of ADAPT[2] can be found at: http://adapt2.sis.pitt.edu/wiki/ADAPT2.

References

Bower M. (2008) A taxonomy of task types in computing. In *13th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE'2008*, June 30–July 2, 2008 (ed. M. Bower), pp. 281–285. ACM Press, Madrid.

Brusilovsky P. (2001) Adaptive hypermedia. *User Modeling and User-Adapted Interaction* **11**, 87–110.

Brusilovsky P. & Eklund J. (1998) A study of user-model based link annotation in educational hypermedia. *Journal of Universal Computer Science* **4**, 429–448.

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

Brusilovsky P. & Pesin L. (1994) An intelligent learning environment for CDS/ISIS users. In *The Interdisciplinary Workshop on Complex Learning in Computer Environments (CLCE94)* May 16–19, 1994 (eds J.J. Levonen & M.T. Tukianinen), pp. 29–33. EIC, Joensuu Available at: http://cs.joensuu.fi/~mtuki/www_clce.270296/Brusilov.html.

Brusilovsky P. & Sosnovsky S. (2005a) Individualized exercises for self-assessment of programming knowledge: an evaluation of QuizPACK. *ACM Journal on Educational Resources in Computing* **5**, Article No. 6.

Brusilovsky P. & Sosnovsky S. (2005b) Engaging students to work with self-assessment questions: a study of two approaches. In *10th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE'2005*, June 27–29, 2005 In: ACM Press, Monte de Caparica. pp. 251–255.

Brusilovsky P., Sosnovsky S. & Shcherbinina O. (2005) User modeling in a distributed e-Learning architecture. In *10th International Conference on User Modeling (Um'2001)* (eds L. Ardissono, P. Brna & M. Antonija), pp. 387–391. Berlin: Springer Verlag, Edinburgh.

Brusilovsky P., Sosnovsky S. & Yudelson M. (2006) Addictive links: the motivational value of adaptive link annotation in educational hypermedia. In *4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (Ah'2006)* June 21–23, 2006 (eds V. Wade, H. Ashman & B. Smyth), pp. 51–60. Springer Verlag, Dublin.

Davidovic A., Warren J. & Trichina E. (2003) Learning benefits of structural example-based adaptive tutoring systems. *IEEE Transactions on Education* **46**, 241–251.

Douce C., Livingstone D. & Orwell J. (2005) Automatic test-based assessment of programming: a review. *ACM Journal on Educational Resources in Computing* **5**, Article No. 4.

Fischer S. & Steinmetz R. (2000) Automatic creation of exercises in adaptive hypermedia learning systems. In *Eleventh ACM Conference on Hypertext and Hypermedia (Hypertext 2000)* May 30–June 3, 2000, pp. 49–55. ACM Press, San Antonio, TX.

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*, (eds S. Lobodzinski & I. Tomek), pp. 634–636. AACE, Toronto.

Hsiao I., Brusilovsky P. & Sosnovsky S. (2008) Web-based parameterized questions for object-oriented programming. In *E-Learn 2008* November 17, 2008 (eds C.J. Bonk, M.M. Lee & T. Reynolds), pp. 3728–3735. AACE, Las Vegas, NV.

Kashy E., Thoennessen M., Tsai Y., Davis N.E. & Wolfe S.L. (1997) Using networked tools to enhance student success rates in large classes. *27th ASEE/IEEE Frontiers in Education Conference*. Pittsburgh.

Kavcic A. (2004) Fuzzy user modeling for adaptation in educational hypermedia. *IEEE Transactions on Systems, Man, and Cybernetics* **34**, 439–449.

Koffman E.B. & Perry J.M. (1976) A model for generative CAI and concept selection. *International Journal on the Man-Machine Studies* **8**, 397–410.

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**, 34–43.

Kumar A.N. (2000) Dynamically generating problems on static scope. In *The Fifth Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE 2000*, July 2000 (ed. B. Manaris), pp. 9–12. ACM Press, Helsinki.

Kumar A. (2005a) Generation of problems, answers, grade and feedback – case study of a fully automated tutor. *ACM Journal on Educational Resources in Computing* **5**, Article No. 3.

Kumar A. (2005b) Rule-based adaptive problem generation in programming tutors and its evaluation. In *Workshop on Adaptive Systems for Web-Based Education at 12th International Conference on Artificial Intelligence in Education, AI-Ed'2005*, July 18, 2005 (eds P. Brusilovsky, R. Conejo & E. Millán), pp. 35–43. IOS Press, Amsterdam.

Kumar A.N. (2006) A scalable solution for adaptive problem sequencing and its evaluation. In *4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (Ah'2006)*, June 21–23, 2006 (eds V. Wade, H. Ashman & B. Smyth), pp. 161–171. Springer Verlag, Dublin.

Malmi L., Karavirta V., Korhonen A. & Nikander J. (2005) Experiences on automatically assessed algorithm simulation exercises with different resubmission policies. *ACM Journal on Educational Resources in Computing* **5**, Article No. 7.

Martin B. & Mitrovic A. (2002) Automatic problem generation in constraint-based tutors. In *6th International Conference on Intelligent Tutoring Systems (Its'2002)* (eds S.A. Cerri, G. Gouardères & F. Paraguaçu), pp. 388–398. Berlin: Springer-Verlag, Biarritz.

Mayo M. & Mitrovich A. (2000) Using a probabilistic student model to control problem difficulty. In *5th International Conference on Intelligent Tutoring Systems (Its'2000)* (eds G. Gauthier, C. Frasson & K. Vanlehn), pp. 524–533. Berlin: Springer-Verlag, Montreal.

Merat F.L. & Chung D. (1997) World Wide Web approach to teaching microprocessors. In *Frontiers in Education Conference*, pp. 838–841. Stipes Publishing L.L.C., Pittsburgh, PA.

Mitrovic A. (2007) Evaluating the effect of open student models on self-assessment. *International Journal of Artificial Intelligence in Education* **17**, 121–144.

Mitrovic A. & Martin B. (2004) Evaluating adaptive problem selection. In *Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (Ah'2004)* (eds P. De Bra & W. Nejdl), pp. 185–194. Berlin: Springer-Verlag, Eindhoven.

Myller N. (2006) Automatic prediction question generation during program visualization. In *Fourth Program Visualization Workshop* June 29–30, 2006 (ed. G. Rösslin), pp. 89–93. Elsevier B.V., Amsterdam.

Olston C. & Chi E.H. (2003) Scenttrails: integrating browsing and searching on the Web. *ACM Transactions on Computer-Human Interaction* **10**, 177–197.

Sitthisak O., Gilbert L. & Davis H.C. (2008) Deriving e-assessment from a competency model. In *Eighth IEEE International Conference on Advanced Learning Technologies* July 1st–July 5th (eds J. Cordeiro, J. Filipe & S. Hammoudi), pp. 327–329. Santander, Cantabria.

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

Ullrich C., Lu T. & Melis E. (2009) A new framework for dynamic adaptations and actions. In *4th European Conference on Technology Enhanced Learning (ECTEL 2009)* September 29–October 2, 2009 (eds U. Cress, V. Dimitrova & M. Specht), pp. 67–72. Springer-Verlag, Nice.