KnowledgeZoom for Java: A Concept-Based Exam Study Tool with a Zoomable Open Student Model

Peter Brusilovsky
University of Pittsburgh
Pittsburgh, USA
peterb@pitt.edu

Dhruba Baishya
ON24, Inc.
San Francisco, CA
dhruba.baishya@on24.com

Roya Hosseini
University of Pittsburgh
Pittsburgh, USA
roh38@pitt.edu

Julio Guerra
Universidad Austral de Chile
Valdivia, Chile
jguerra@inf.uach.cl

MinEr Liang
Fudan University
Shanghai, China
michelle_liang@fudan.edu.cn

***Abstract*-- This paper presents our attempt to develop a personalized exam preparation tool for Java/OOP classes based on a fine-grained concept model of Java knowledge. Our goal was to explore two most popular student model-based approaches: open student modeling and problem sequencing. The result of our work is a Java exam preparation tool, KnowledgeZoom. The tool combines an open concept-level student model component, Knowledge Explorer and a concept-based sequencing component, Knowledge Maximizer into a single interface. This paper presents both components of KnowledgeZoom, reports results of its evaluation, and discusses lessons learned.**

Keywords-Problem Sequencing, Open Student Modeling, Progressive Zoom

I. Introduction

Exam preparation is a challenging task for college students. Within a short period of time, typically a week or less, a student needs to review the content that was studied over the whole semester, identify possible knowledge gaps and misconceptions, and fill these gaps. A personalized learning tool based on a long-term student model could be very helpful in this process. By reflecting students' progress over the whole semester, a student model can distinguish topics that were learned and need just a quick refresh of topics that were missed and may need a thorough review. Using this model, a personalized exam preparation tool can individually guide each student through the study process.

Surprisingly, we were not able to find any attempt to develop a personalized exam preparation tool. While a range of personalized sequencing and adaptive navigation approaches have been developed (see Section II), all approaches known to us are focused on supporting regular learning process that guides students through the whole process of subject learning starting at the very beginning. In our past work, we explored a number of personalized guidance approaches. In particular, we developed several systems to support personalized guidance for a course on Java and Object Oriented Programming (OOP) including a topic-based guidance system JavaGuide [8] and a social guidance system Progressor+ [7]. While these tools were highly efficient in guiding student practice over the duration of the course, we found that the guidance provided by either of them is not sufficient for exam preparation. Neither coarse-grained topic-based guidance, nor social guidance was able to recognize specific holes in students' knowledge and to offer the best way to bridge the gap. The experience with both tools caused us to believe that an exam preparation tool requires a fine-grained concept-level student model and a specific gap-focused guidance approach.

This paper presents our attempt to develop an exam preparation tool for Java/OOP classes based on a fine-grained concept model of Java knowledge. Our goal was to explore two most popular student model-based personalized guidance approaches: open student modeling and problem sequencing. The idea of open student modeling is to show the state of a student model to the student in order to help her reflect on her knowledge, identify gaps, and focus on filling these gaps. The idea of adaptive problem sequencing is to generate a personalized sequence of problems that will help the student to efficiently practice her missing knowledge. The Java exam preparation tool KnowledgeZoom (KZ) that we developed combines an open concept-level student model component Knowledge Explorer (KE) and a concept-based sequencing component Knowledge Maximizer (KM) in a single interface. This paper presents both components of KZ focusing on the challenges of concept-level open student modeling and sequencing, reports its evaluation, and discusses lessons learned.

II. Related Work

A. Open Student Modeling

Open student modeling is an important research direction in the area of intelligent educational systems. Unlike the mainstream research in this area that use a student model as a hidden information source to adapt the learning process to students' needs, open student modeling researchers argue that a student model has its own pedagogical value and should be visible and editable by students. A range of benefits have been reported on opening the student models to the learners, such as increasing the learner's awareness of the developing knowledge, difficulties and the learning process, and students' engagement, motivation, and knowledge reflection [4; 13; 16]. Visual presentations of the student model vary from displaying high-level summaries (such as skill meters) [13] to complex concept maps or Bayesian Networks [16]. In particular, several projects explored TreeMaps [14] as a way to present hierarchical student models [2; 5; 10; 11]. Yet, the student models explored in earlier projects were relatively simple and typically presented in one-shot that eliminated a need to explore the model in detail. In contrast, our work focuses on reasonably complex concept-based user models with hundreds of concepts and studies a progressive zoom [10] approach to explore these models.

978-0-7695-5009-1/13 $26.00 © 2013 IEEE
DOI 10.1109/ICALT.2013.86

IEEE computer society

B. Adaptive Problem Sequencing

Adaptive problem sequencing is one of the oldest technologies in the area of intelligent educational systems. The goal of this technology is to generate a personalized sequence of problems for every student so that they can achieve their learning goal in a most optimal way. A range of approaches were proposed for adaptive problem sequencing including approaches based on associative mechanisms [9], dynamic problem difficulty [12], and metadata [6]. Concept-based problem sequencing [1] is a subclass of sequencing approaches. It is based on a fine-grained concept-level domain model that is used to index problems.

Although all sequencing approaches try to find the most optimal problems for the students, they might fail when the user model is incorrect. In such cases the system selection cannot be relied upon and students should be able to select the problems themselves. In our previous interfaces for accessing learning content for Java, we have tried to reduce the negative effects of sequencing errors through adaptive navigation support technologies that do not force the students to work on a problem considered the best by the sequencing mechanism, but provide annotation-based navigation support that combines intelligent guidance with human decision-making [3; 8]. In this paper, we return to a more traditional problem sequencing mechanism that we consider as a promising approach in the exam preparation context when time is limited and an optimal guidance becomes quite critical.

III. The KnowledgeZoom (KZ) Study Tool

To investigate the value of concept-based personalization in the context of exam preparation, we developed a concept-based exam study tool KZ. The goal of KZ is to help the students identify their course knowledge gaps and provide tools to bridge these gaps in an effective way. The first part of this dual goal is supported by the KE component, a concept-based hierarchical zoomable open student model. The second goal is supported by the KM, a concept-based adaptive problem sequencing tool. The interface of KZ (Fig. 1) provides direct access to the KE model and a button to launch the KM. Students access the tool through a personalized learning portal along with several other study tools such as JavaGuide [8] and Progressor+ [7].

A. The Domain Model and the Learning Content

KZ is based on a concept-level model of knowledge about Java and OOP. This model is formed by a subset of concepts from the Java ontology http://www.sis.pitt.edu/~paws/ont/java.owl built by the PAWs lab. The Java ontology includes 344 concepts organized into an 8-level tree. The learning content in KZ is formed by 103 parameterized self-assessment questions that were developed in our team as a part of an earlier project [8]. Each question is indexed with ontology concepts. The indexing classifies the *prerequisite* concepts that should be known before approaching the question and the *outcome* concepts to be mastered by working with the question. The number of concepts associated with a single question ranges from 5 to 52 (0 to 41 prerequisites, 1 to 12 outcomes). These questions cover the 188 most important concepts of Java which form the KZ *domain model*.

B. The Knowledge Explorer (KE)

KE is a multi-level open student model visualized with a zoomable Treemap. The information presented by KE is an overlay model of Java Knowledge based on the KZ ontological domain model. The overlay student model in KZ is maintained by a user modeling service, PERSEUS [15], which updates the model after every attempt to answer a question and changes the knowledge level of concepts related to the question.

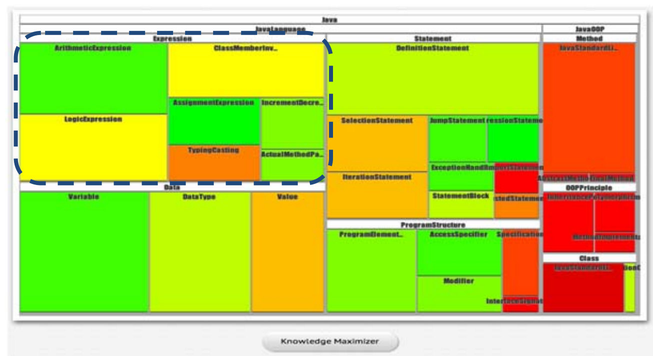


Figure 1. The KnowledgeZoom interface showing the top level of the Knowledge Explorer map and a button to launch Knowledge Maximizer.

Figure 2. Zooming on the node *Expressions* (top left corner as marked in Fig. 1) reveals next level of the concept hierarchy. Now the user can see that the node LogicExpression that has intermediate knowledge as a whole (shown as yellow) consists of several well learned and several unknown concepts.

A zoomable Treemap was selected to present the student model due to its relatively large size and hierarchical nature. The Treemap layout shows only four levels of concept hierarchy starting from the current top node and hiding lower-level nodes behind its ancestor node. The user, however, can zoom in any node. After zooming in, the node expands becoming the top node and occupying the whole view. Zooming-in immediately exposes previously hidden levels of hierarchy. For example, Fig. 2 shows the results of zooming into a second level concept, *Expression* shown in the top left quadrant of Fig. 1.

In the Treemap layout, each node (a concept in the Java ontology) is shown as a colored rectangle. A leaf concept of the ontology corresponds to a terminal node of the Treemap.

The *size* of a node represents the importance of a concept in the context of Java language and its chance to be checked as part of the exam. We measure it by counting how many questions are related to the leaf concept corresponding to this leaf node in the Treemap. Since the number of exercises related to nodes can be quite different, which leads to a large difference in the node sizes, we use the *log*$_2$*(size)* to moderate the differences. The *color* of a node represents the level of concept knowledge demonstrated by a student. We use 10 colors from red to green to represent the progression from weaker to stronger knowledge. In a hierarchical zoomable layout, a leaf node directly represents the importance and knowledge level of a concept with its size and color respectively, while each intermediate node accumulatively aggregates importance and concept knowledge from its child nodes. As a result of the aggregation, the upper-level views show overviews of students' state of knowledge on higher levels (Fig. 1), while being able to explore detailed knowledge of every concept as zooming into lower levels of the ontology (Fig. 2). The calculation of the aggregated size and color is important to bridge the gaps between lower and higher levels of views. In KE, the size aggregation is provided by Treemap. For the color aggregation, the color of an intermediate node is the average color of its direct child nodes weighted with their sizes in order to reflect the importance of the associated concepts.

C. The Knowledge Maximizer (KM)

The goal of the KM is to provide the learner with a set of questions, which will help her achieve her learning goals by recommending the questions with the highest gain. KM considers the following factors for selection of the best activities which are considered as questions:

How much is the student prepared to do the activity? The students should be prepared to do the proposed activities. The activities for which the student has low levels of knowledge of prerequisite concepts are not good suggestions. We calculate the learner knowledge for each of the prerequisite concepts of an activity to see how much the student is prepared to do it. Equation (1) shows the formula:

$$K = \frac{\sum_{i}^{M_r} k_i w_i'}{\sum_{i}^{M_r} \max(k_i) w_i'} \qquad w_i' = \log(w_i) \tag{1}$$

where K is the level of the learner's knowledge in the prerequisites of the activity; w_i' is the smoothed weight for the activity-concept (we do it by performing log function on the weight); k_i is the level of the learner's knowledge in the $i-$th concept and M_r is the set of prerequisite concepts for the activity. Higher knowledge of prerequisite concepts of an activity (larger K) makes it a better candidate to be selected by the optimizer.

What is the impact of the activity? The formula for this impact is shown as (2):

$$I = \frac{\sum_{i}^{M_o} w_i'(1-k_i)}{\sum_{i}^{M_o} w_i'} \tag{2}$$

where M_o is the set of concepts of the outcome of the activity. Impact I for a certain activity shows that when the activity has higher impact and hence it will be a better candidate to be selected by the optimizer.

Has the user already completed the activity? We use success rate to understand how much the learner has learned from an activity. We define it as (3):

$$\bar{S} = 1 - \frac{s}{t+1} \tag{3}$$

where $\bar{S}$ is the inverse success rate of the student in the activity; s is the number of the times the student has succeeded in the activity; and t is the total number of times the student has tried the activity

Having calculated the above factors, we can simply rank the activities using (4):

$$R = \alpha K + \beta I + \gamma \bar{S} \tag{4}$$

where R is the rank of the activity and α, β, γ are the weights assigned to each of the above mentioned factors respectively.

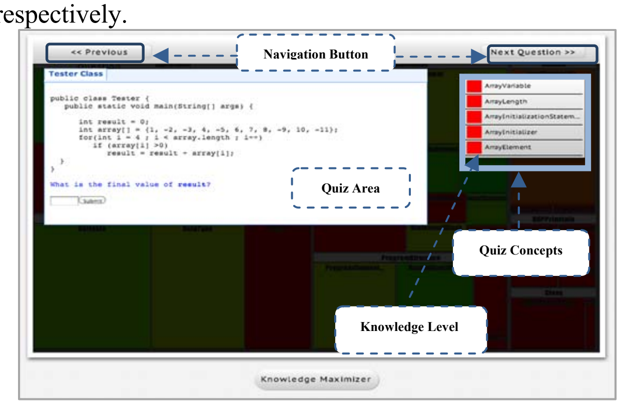


Figure 3. The Knowledge Maximizer interface

Fig. 3 shows the interface of KM. The list of concepts covered by the quiz is also shown on the right side of this panel. The color next to each concept represents the student's current knowledge level.

IV. THE EVALUATION

To assess the value of KZ we conducted a classroom study in the context of a Java-based undergraduate course *Introduction to Object Oriented Programming* at the School of Information Sciences, University of Pittsburgh. All students enrolled in this course were invited to use the KZ for the final exam preparation. The study started on December 4th 2012 about a week before the final exam. Note that the class also used QuizGuide and Progressor+ to access Java questions that were available from the beginning of the semester. As a result, many students learned a considerable number of Java concepts by the time they started with KZ and were able to benefit from the "gap filling" nature of the system. Figure 1 showed how knowledge map might have looked to a typical student during the first session of KZ – many concepts were learned, yet there were still many orange and red gaps to fill.

A. Log Analysis

We hypothesized that KZ bridges the existing gap in the student's knowledge by recommending a set of questions that bring a student to a better level of knowledge. To examine our hypothesis, we considered the following system usage parameters:

- *Attempts* (the total number of questions attempted)
- *Success Rate* (the percentage of correctly answered questions)
- *Distinct Questions* (the number of distinct attempted questions)
- *Attempts per question* (the number of attempts for doing a question)
- *Sessions* (the number of sessions the students worked with the systems)

In our analysis we separately counted question accesses from KZ and questions accessed from either QuizGuide/Progressor+. Attempts made from KZ were made by 14 students while attempts made from QuizGuide/Progressor+ were made by 17 students. As can be seen in Table I, the total number of attempts made from QuizGuide/Progressor+ was much larger, which is natural since the students were familiar with QuizGuide and Progressor+ from the beginning of the class. Yet, it is quite remarkable that KZ, which was introduced just a week before the exam, was considerably used. We also observed that KZ presented students with interesting and challenging questions as shown by the increase of attempts per question.

TABLE I. SYSTEM USAGE SUMMARY

Parameter	KZ (n=14)	QG,P+ (n=17)
Attempts	434	3245
Success rate	58%	64%
Distinct questions	119 (27%)	1145 (35%)
Attempts per questions	3.64	2.83
Attempt per Sessions	10.58	21.35

KZ = KnowledgeZoom; QG = QuizGuide; P+ = Progressor+.

To assess whether KM was successful in "maximizing" students' steps towards the goals, we grouped questions into three different complexity levels based on the number of involved concepts (Easy, Moderate and Complex) [8]. A question with 15 or fewer concepts is considered to be Easy, 16 to 90 as Moderate, and 90 or higher as Complex. Table II lists the number of attempts made to easy, moderate, and complex questions from KZ and from QuizGuide/Progressor+. The data revealed that although in KZ the fraction of easy/moderate question attempts was smaller than in QuizGuide/Progressor+, the number of attempts to complex questions which helped students reach their goal faster by covering many concepts at once was about 2.5 times greater. Another interesting result was that despite a remarkable increase in complex questions, the success rates across all systems were comparable.

TABLE II. NUMBER OF ATTEMPTS, SUCCESS RATES BY SYSTEM AND COMPLEXITY LEVEL

Complexity	KZ (n=14)		QG,P+ (n=17)	
	Number of Attempts	Success rate	Number of Attempts	Success rate
Easy	27 (6.2%)	93%	1123 (34.6%)	73%
Moderate	189 (43.5%)	68%	1471 (45.3%)	61%
Complex	218 (50.2%)	46%	651(20.1%)	55%
Total	434	58%	3245	64%

KZ = KnowledgeZoom; QG = QuizGuide; P+ = Progressor+.

B. Student Feedback Analysis

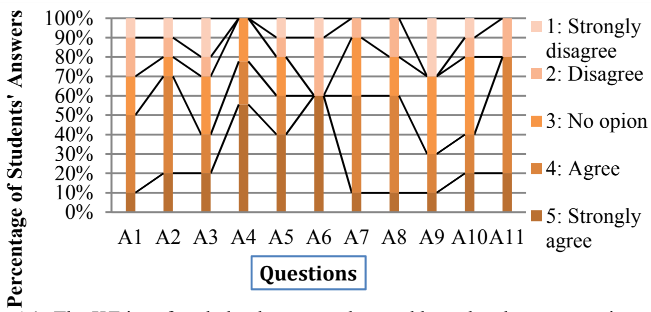
At the end of the evaluation, students were asked to provide feedback about KZ and other systems used in the course. Of 21 students who returned the forms, 11 students used KZ, however only 10 of them answered questions related to KZ. Since KZ is the focus of this paper, the following analysis is based on the KZ part of the questionnaire and analyzes the answers of these 10 students.

The results are shown in Fig 4. Overall, 80% of the students considered the KZ system helpful as a whole (A11), which suggests that it is helpful to combine the two individual components, KE and KM together. For KE, 70% considered its interface helpful to identify their knowledge weak points (A2), which provides evidence to support the main goal of KE. 60% agreed that use of color for Treemap nodes to show their concept knowledge was clear (A5), and 60% agreed that the use of color aggregation to show their higher-level concept knowledge was clear (A6); 60% agreed that the use of Treemap node size to show concept importance was clear (A7), and 60% agreed that the use of size aggregation to show the importance of higher-level concepts was clear (A8). We need to investigate these results further.

For KM, about 78% of the students considered the ability of KM to generate quizzes that cover many concepts as helpful (A4)[1], which provides evidence to support the main goal of KM. Only 30% noted that the quizzes generated by KM were too simple for them (A9), supporting the log analysis data that KM challenged the students. However, only 40% considered that the KZ interface helped them to access the most relevant quizzes (A3), and only 40% considered that the KZ system accelerated their preparation for the final exam (A10).

[1] Only nine students answered this question.

The analysis of student feedback indicated that many students were frustrated that the questions provided by KM component were not affected by their KE zooming activity. They expected that zooming into a specific difficult concept should allow them to access to questions specifically related to that concept.



A1: The KZ interface helped me to understand how the class content is organized. (1 2 3 4 5)
A2: The KZ interface helped me to identify my weak points. (1 2 3 4 5)
A3: The KZ interface helped me to access the most relevant quizzes. (1 2 3 4 5)
A4: The ability of the Knowledge Maximizer to generate quizzes that cover many concepts was helpful. (1 2 3 4 5)
A5: The use of color for Treemap nodes to show my concept knowledge was clear. (1 2 3 4 5)
A6: The use of color aggregation to show my higher-level concept knowledge is clear. (1 2 3 4 5)
A7: The use of Treemap node size to show concept importance is clear. (1 2 3 4 5)
A8: The use of size aggregation to show the importance of higher-level concepts is clear. (1 2 3 4 5)
A9: The quizzes generated by the Knowledge Maximizer were too simple for me. (1 2 3 4 5)
A10: The KZ system accelerated my preparation for the final exam. (1 2 3 4 5)
A11: The KZ system as a whole has been helpful. (1 2 3 4 5)

Figure 4. Subjective evaluation: questions and results

V. Conclusion and Future Work

In this paper, we have explored two concept-based approaches - an open zoomable student model and adaptive problem sequencing – to support students to prepare for their final exams in a Java programming class. The results of our study showed that our tool attracted student attention and was recognized by them as considerably helpful in visualizing their Java knowledge and in revealing knowledge gaps. KZ was able to generate challenging questions that shortened the path to students' learning goals. In our future work we plan to improve KZ and implement better connections between its components by integrating concept zooming and question access; and to further investigate how to represent more clearly the users' knowledge with the Treemaps attributes (color and size).

Acknowledgment

This research was supported in part by the National Science Foundation under Grant No. 0447083. Julio Guerra is supported by a Chilean Scholarship (Becas Chile) from the National Commission for Science Research and Technology (CONICYT, Chile) and the Universidad Austral de Chile. MinEr Liang was a Visiting Scholar at the School of Information Science, University of Pittsburgh when she worked on this project.

References

[1] P. Brusilovsky, A framework for intelligent knowledge sequencing and task sequencing. in: Proc. Second International Conference on Intelligent Tutoring Systems, ITS'92 C. Frasson, G. Gauthier and G. McCalla, eds.,Springer-Verlag, Montreal, Canada, 1992, pp.499-506.

[2] P. Brusilovsky, I.-H. Hsiao and Y. Folajimi, QuizMap: Open Social Student Modeling and Adaptive Navigation Support with TreeMaps. in: Proc. 6th European Conference on Technology Enhanced Learning (ECTEL 2011) Lecture Notes in Computer Science 6964, Springer-Verlag, 2011, pp.71-82.

[3] P. Brusilovsky and S. Sosnovsky, Engaging students to work with self-assessment questions: A study of two approaches. in: Proc. 10th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE'2005 2005, pp.251-255.

[4] S. Bull, Supporting learning with open learner models. in: Proc. 4th Hellenic Conference on Information and Communication Technologies in Education A, Athens, Greece, 2004, pp.47-61.

[5] M.H. Falakmasir, I.-H. Hsiao, L. Mazzola, N. Grant and P. Brusilovsky, The Impact of Social Performance Visualization on Students. in: Proc. 12th IEEE International Conference on Advanced Learning TechnologiesIEEE, Rome, Italy, 2012, pp.565-569.

[6] S. Fischer, Course and exercise sequencing using metadata in adaptive hypermedia learning systems. ACM Journal on Educational Resources in Computing 1(1) (2001).

[7] I.-H. Hsiao and P. Brusilovsky, Motivational Social Visualizations for Personalized E-Learning. in: Proc. 7th European Conference on Technology Enhanced Learning (EC-TEL 2012) Lecture Notes in Computer Science 7563, Saarbrücken, Germany, 2012, pp.153-165.

[8] I.-H. Hsiao, S. Sosnovsky and P. Brusilovsky, Guiding students to the right questions: adaptive navigation support in an E-Learning system for Java programming. Journal of Computer Assisted Learning, 26(4) (2010) 270-283.

[9] A.N. Kumar, A Scalable Solution for Adaptive Problem Sequencing and its Evaluation. in: Proc. 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems Lecture Notes in Computer Science 4018, Springer, 2006, pp.161-171.

[10] M. Liang, J. Guerra, G.E. Marai and P. Brusilovsky, Collaborative E-Learning through Open Social Student Modeling and Progressive Zoom Navigation. in: Proc. COLLABORATECOM 2012 - 8th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, Pittsburgh, USA, 2012.

[11] S.N. Lindstaedt, G. Beham, B. Kump and T. Ley, Getting to Know Your User – Unobtrusive User Model Maintenance within Work-Integrated Learning Environments. in: Proc. 4th European Conference on Technology Enhanced Learning (ECTEL 2009) Springer-Verlag, Nice, France, 2009, pp.73-87.

[12] A. Mitrovic and B. Martin, Evaluating adaptive problem selection. in: Proc. Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2004) Lecture Notes in Computer Science 3137, Springer-Verlag, Berlin, 2004, pp.185-194.

[13] A. Mitrovic and B. Martin, Evaluating the Effect of Open Student Models on Self-Assessment. International Journal of Artificial Intelligence in Education, 17(2) (2007) 121-144.

[14] B. Shneiderman, Tree visualization with tree-maps: 2-d space-filling approach. ACM Transactions on Graphics, 11(1) (1992) 92-99.

[15] M. Yudelson, Providing service-based personalization in an adaptive hypermedia system. PhD Thesis. U. of Pittsburgh, 2010.

[16] J.-D. Zapata-Rivera and J.E. Greer, Interacting with Inspectable Bayesian Student Models. International Journal of Artificial Intelligence in Education, 14(1) (2004) 1-37.