2014

Personalizing Software Development Practice Using Mastery-based Coaching

Chris BOESCH
Singapore Management University, cboesch@smu.edu.sg

Sandra BOESCH
Singapore Management University, sandraboesch@smu.edu.sg

Personalizing Software Development Practice Using Mastery-based Coaching

Chris Boesch, Sandra Boesch
Singapore Management University, Pivotal Expert
Singapore

Abstract

The authors previously developed a system to facilitate the self-directed learning and practicing of software languages in Singapore. One of the goals of this self-directed learning was to enable the development of student mentors who would then be able to assist other students during classroom sessions. Building on this work, the authors extended the platform to support personalized coaching with the goals of further enabling and preparing students to mentor their peers. This paper covers the challenges, insights, and features that were developed in order to develop and deploy this mastery-based coaching feature.

1. Introduction

Personalized, one-to-one teaching and mentoring has been recognized as one of the most effective methods of maintaining student engagement and enhancing learning [1][2]. The authors have been experimenting with ways to scale-up improved personalized assessment and learning using cloud-based systems. In 2011, the authors invited nearly 200 students who were on the borderline of being rejected for university admission to participate in a new program, which would give them a second chance to gain one of the coveted admission spots. These students were asked to learn basic Java and Python programming languages on their own within a two week time period. The students were instructed to use SingPath [3], the tool developed by the authors for practicing programming in a self-directed manner. The students were expected to demonstrate minimal competence, and then take part in a short software tournament [4].

Since this trial, the authors have extended the platform to support software lab delivery in classroom settings [5]. Additionally, SingPath has been extended to support added personalized, self-directed learning in preparation for classroom sessions [6]. Lastly, the authors implemented an automated-mentor assignment feature [7] which automates the process of identifying students that have mastered material and pairs them up to serve as mentors to students still working on assignments. In 2014, the authors designed and outlined how the system used to support self-directed learning and peer-based mentoring in a classroom environment could be extended to identify and coach students on a national scale [8]. After observing how students were preparing for classroom sessions and national software development tournaments, the authors identified the need and opportunity to develop an automated coaching feature to improve the efficiency of time spent resolving previously solved problems.

2. SingPath

SingPath is a web-based tool that enables users to practice programming in several software languages within a gaming environment. The platform started as a tool to provide students with immediate online feedback on solutions to programming problems and expanded over time to support different types of blended learning needs for a variety of classes and classroom settings. A picture of the interface is shown below in Figure 1.

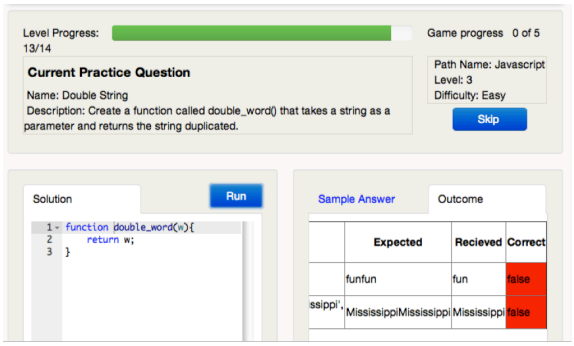


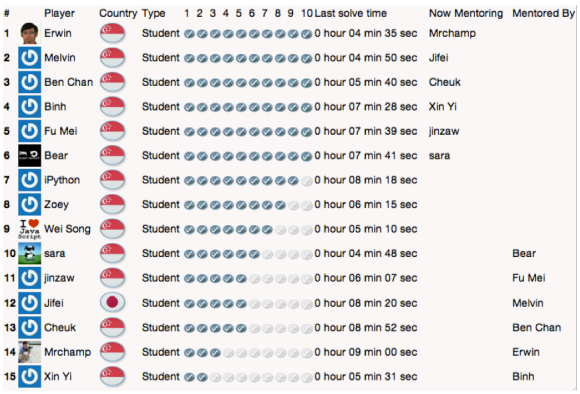
Figure 1: Normal Gameplay

The SingPath platform supports traditional self-directed learning mechanisms such as badges and completion metrics as well as features for use in classrooms, such as tournaments. In these software tournaments, participants are usually asked to solve around ten short programming problems. SingPath

tournaments are non-graded exercises based on serious gaming [9], which provide the student with an opportunity for hands on learning and present the instructor with an opportunity for data collection and customization of classroom materials.

3. Tournaments

During software tournaments, a live ranking is usually displayed on a screen to show the class progress. This ranking provides feedback to the students regarding how the participants are progressing and provides the tournament facilitator with information as to which participants may need the most assistance as well as how long the class is likely to take to finish the tournament (Figure 2).



#	Player	Country	Type	1	2	3	4	5	6	7	8	9	10	Last solve time	Now Mentoring	Mentored By
1	Erwin		Student											0 hour 04 min 35 sec	Mrchamp	
2	Melvin		Student											0 hour 04 min 50 sec	Jifei	
3	Ben Chan		Student											0 hour 05 min 40 sec	Cheuk	
4	Binh		Student											0 hour 07 min 28 sec	Xin Yi	
5	Fu Mei		Student											0 hour 07 min 39 sec	jinzaw	
6	Bear		Student											0 hour 07 min 41 sec	sara	
7	IPython		Student											0 hour 08 min 18 sec		
8	Zoey		Student											0 hour 06 min 15 sec		
9	Wei Song		Student											0 hour 05 min 10 sec		
10	sara		Student											0 hour 04 min 48 sec		Bear
11	jinzaw		Student											0 hour 06 min 07 sec		Fu Mei
12	Jifei		Student											0 hour 08 min 20 sec		Melvin
13	Cheuk		Student											0 hour 08 min 52 sec		Ben Chan
14	Mrchamp		Student											0 hour 09 min 00 sec		Erwin
15	Xin Yi		Student											0 hour 05 min 31 sec		Binh

Figure 2: Live Onscreen Tournament Rankings

While preparing for these tournaments, many students attempt to work ahead by solving more problems in order to increase their mastery and to prepare themselves for future tournaments. However, once a student has solved 50 to 100 programming problems they often desire to revisit previous material to increase their mastery of that material and to better prepare themselves, should similar problems appear in upcoming tournaments.

Because SingPath is designed with paths and levels, each student must pick a path or programing language such as Python, Java, or Ruby and solve a level, which consist of questions on specific topics such as functions, conditions, or strings. Usually this revisiting or review process entailed resolving the problems on a given level. This was an inefficient process since students would need to resolve all problems in that level regardless of how well they had performed when solving the problems the first time. What students really wanted was a way to practice and resolve the problems where they had been the least proficient.

4. Relative Performance

As players solve problem while playing SingPath, SingPath keeps track of both the player's performance and the performance of all other players that have solved a given problem. This enables SingPath to determine a percentile ranking of where the player's performance for a problem falls in relation to all other players. In addition to the relative speed performance in which a player solved a problem, SingPath also tracks the number of attempts and lines of code that a player required to solve a problem. As such, SingPath compares each student's performance with all other players. With this data, SingPath is then able to create a prioritized list of problems that the student can resolve starting with the problem with the worst relative performance and working up from there. SingPath is then able to present these problems to the player one at a time enabling players to focus on the problems where they demonstrated the most opportunity for improvement compared with their peers.

5. Mastery-based Coaching

The mastery-based coaching feature and four initial coaching personas were launched in February 2014 to support the self-directed learning of hundreds of Singaporean students who would be preparing for and competing in upcoming national coding tournaments [8]. The authors approach to coaching was designed to be scalable. As long as more than one player has attempted a problem and the current player has attempted at least one problem, the coaching feature should be able to prioritize a list of problems for the player to resolve.

As players resolve problems, their proficiency in solving these problems is likely to improve. Since the list of problems is prioritized by presenting the players lowest percentile performance first, this means that the player was able solve all other problems at higher relative performance. Considering that the player has demonstrated better relative performance for all other problems, the player is usually able to resolve their least proficient problem a little better when it receives their focused attention. Added to this is the fact that in order to be coached to resolve a problem, players must have solved the problem at least once previously. The authors observed that players are better able to solve problems in coaching sessions since the problems being presented have been solved before and the problems presented are the ones most likely to fall below the player's demonstrated relative level of performance.

As players improve their proficiency on more problems, their level of demonstrated proficiency on their least proficient problem increases. Once the players' least demonstrated proficiency rises above

some target level of proficiency, the players are encouraged by the coaches to stop resolving problems and to go and practice new problems. The target level of proficiency for the original prototype was set to be 80%. This 80% target was chosen to align coaching targets with the authors' research into identifying and training mentors.

In order to make this process more interesting than simply resolving problems, the authors implemented the new coaching feature using a collection of coaching personas. The four initial coaching personas were chosen to provide players with choices based on the type of coaching, feedback, and encouragement that they would want to receive.

One of the coaches, named Shannon, is a young girl who is most interested in how the student will feel the next time that he or she has to code with friends and other classmates. Shannon wants the student to be comfortable and confident in his or her coding capability. Shannon makes statements such as: "You are doing great" and "Coding with your friends is so much fun". SGT MJR is a drill sergeant that will use tough love to get the best out of the student's programming practices. The Sandra coach is a recruiter interested in the student doing well in his or her next technical interview where the interviewer is likely to ask the student to write software. And the last coach is Zandar, a young teenage boy who is most interested in seeing the student become "more awesome". Zardar's goal is that the student will be prepared to do well in any upcoming tournaments (See Figure 3).

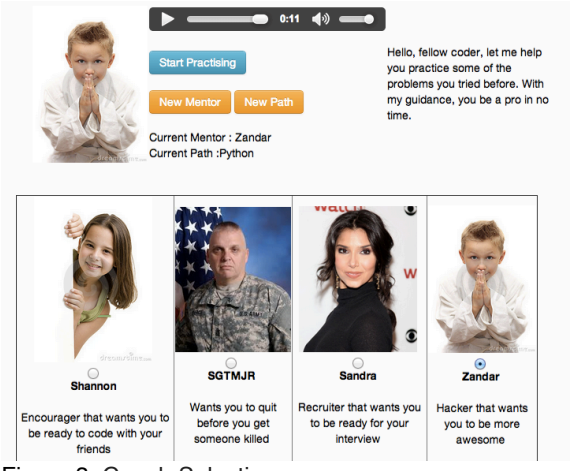


Figure 3. Coach Selection

All of these mentors follow a common pattern of encouraging players to resolve problems in less time or in fewer attempts. Encouragement is provided by playing audio clips along with displaying the text of the coach's comments for the players to read (see Figure 4). The initial implementation involves coaches working down a prioritized list of ten problems based on the player's relative solve times.

SingPath provides the coaching logic with an ordered list of recommended problems for players to reattempt. This ordered list contains players' problem solving data such as previous time, number of attempts, time percentile, and attempts percentile. This information provides the coaching logic with the ability to congratulate players when they resolve problems in less time or fewer attempts than previously achieved. Alternatively, the coach can make comments such as: "We will try that one again later". Having visibility to the attempts data also provides the coaching logic to encourage the player to solve problems in fewer attempts rather than in less time. This format provides some variety in how coaches present problems to resolve. The authors' initial implementation designed the coaches to ask players to solve problems in fewer attempts whenever their attempts percentile was lower than their time percentile. Therefore, coach's feedback provided is based on how many attempts the player requires to resolve a given problem.

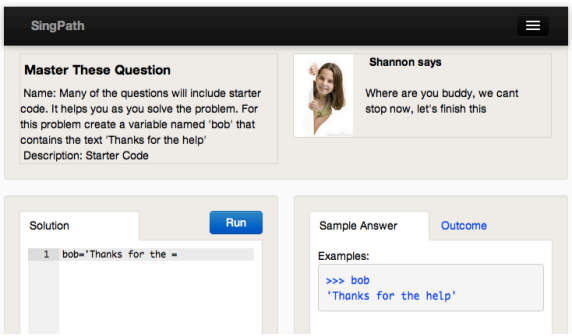


Figure 4. Coaching Gameplay

The addition of coaching is likely to further differentiate the capability and preparedness of students when they come to class sessions or tournaments. Because of the authors' automated mentor assignment approach [7], this increased differentiation is less of a problem. More prepared students will spend less class time coding on their own and more time mentoring other students. Less prepared students will then be assigned a mentor earlier during the sessions. It is also possible that some students who might normally perform in the bottom quartile of the class will have an opportunity to use the coaching feature to enhance their tournament performance if that is their desire.

6. Towards Intrinsic Motivation

Another objective of Mastery-based Coaching has been to avoid providing players with a numeric feedback metric on their progress. Even though players will be increasing their proficiency and relative performance as they resolve problems, they are not shown this numerical and measurable

improvement. The SingPath platform already contains a variety of gamification features such as badges, level completion percentages, event rankings, quest progress, and other items [6]. Rather than tell players that they are at the 57 percentile, coaches simply encourage players to continually get better.

The objective of this design is to endeavor to start moving away from extrinsic motivation factors such as grades, tournaments, and badges towards more intrinsically aligned motivations such as autonomy, mastery, and purpose [10]. As the name implies, Mastery-based Coaching focuses on mastery. The authors would like the players to feel that they are getting better. The coaches focus on conveying and reinforcing that message. For example: "You are faster and more accurate now than you were when you started practicing."

The positive feedback and encouragement provided by the coaches is still an additional extrinsic motivator, but the authors' hypothesis is that by hiding the underlying time and time percentile metrics, players will need to reflect more on how they are doing. The authors hope that by providing status report messages, players might ask themselves questions such as: "How does the coach know that I am getting better" and "Am I really getting better". If SingPath's Mastery-based coaching can encourage players to reflect on their increased mastery of the material and appreciate their progress, this may also lead to additional intrinsic motivation that will encourage more players to practice longer.

7. Conclusion

The Mastery-based coaching method, based on relative performance metrics is an approach put in place to coach students to solve programming language problems faster and in fewer attempts. This platform could be extended to encourage students to use fewer lines of code, use less duplicate code, follow preferred coding conventions, or to use better algorithms. However, due to the authors focus on identifying, developing, and assigning qualified mentors, the authors' research interests continue to focus primarily on speed and on the correlated metric of attempts needed to solve problems quickly and efficiently. In the future, the authors hope to learn from the data generated by students using the Mastery-based coaching feature as they prepare for coding competitions and classroom sessions. The authors plan to adapt any insight from this research for the benefit of programming students world wide.

8. Future Research

As part of the authors' future research on Mastery-based coaching, they have also prototyped the option of having coaches introduce new problems and material rather than systematically directing players to exit coaching sessions to solve new problems. The authors' hypothesis is that some students may enjoy the coaching experience and encouraging environment more than they might enjoy the practice or quest experiences [6]. Staying in the coaching experience might result in the further engagement of these students and their increase in time spent practicing and learning new material. This mode would also provide the benefit of allowing coaches to ensure that players have mastered all material to a given target level of proficiency such as 80% before proceeding on to learning new material. This approach would more closely align with Bloom's approach to mastery-based teaching [1]. In the future, the authors plan to conduct research to determine if having coaches introduce new material leads to better outcomes than simply concluding coaching sessions until material has been attempted using the other options available in SingPath.

9. References

[1] Bloom, B. (1984). "The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring", Educational Researcher, 13:6(4-16).

[2] Leontidis, M., Halatsis, C., & Grigoriadou, M. (2009). MENTORing Affectively the Student to Enhance his Learning. Advanced Learning Technologies, IEEE International Conference.

[3] http://singpath.com

[4] Boesch, C., & Steppe, K. Case Study on Using a Programming Practice Tool for Evaluating University Applicants. 3rd Annual International Conference on Computer Science Education: Innovation and Technology. 2011.

[5] Boesch, C., & Boesch, S. (2012). Tournament-based Teaching. 4th Annual International Conference on Computer Science Education: Innovation and Technology. 2012.

[6] Boesch, C., & Boesch, S. (2013). Adaptive Gameplay for Programming Practice. 5th Annual International Conference on Computer Science Education: Innovation and Technology (CSEIT 2013).

[7] Boesch, C., & Steppe, K. (2014). Automated Mentor Assignment in Blended Learning Environments. 27th Conference on Software Engineering Education and Training (CSEE&T 2014).

[8] Boesch, C., & Boesch, S. (2014). Enabling National Software Competitions to Identify and Enhance Student Mentor Capability in Singapore. Ireland International Conference on Education (IICE-2014).

[9] Adamo-Villani, N., Haley-Hermiz, T., & Cutler, R. (2013). Using a Serious Game Approach to Teach 'Operator Precedence' to Introductory Programming Students. 17th International Conference on Information Visualisation (iV2013).

[10] Park, S., & Kim, C. (2011). Designing a Virtual Tutee System to Enhance College Student Motivation. IEEE 11th International Conference on Advanced Learning Technologies (ICALT 2011).