Interactive Pedagogical Agents for Learning Sequence Diagrams

Sohail Alhazmi(✉), Charles Thevathayan, and Margaret Hamilton

RMIT University, Melbourne, Australia
{sohail.alhazmi,charles.thevathayan, margaret.hamilton}@rmit.edu.au

Abstract. Students struggle to learn sequence diagrams (SDs), as the designs must meet the requirements without violating the constraints imposed by other UML diagrams. Providing manual timely feedback, though effective, cannot scale for large classes. Our pedagogical agent combining data dependencies and quality metrics with rule-based techniques capturing consistency constraints allowed generation of immediate and holistic feedback. The scaffolding approach helped to lower the cognitive overload. The pre- and post-tests and survey results revealed substantially improved learning outcomes and student satisfaction.

Keywords: Pedagogical agent · Constructivism · Interaction diagrams

1 Introduction

A multi-institutional study with 314 participants found that over 80% of graduating students were unable to create a software design or even a partial design [3]. The design and modelling skills are cognitively demanding skills needing formative feedback [10]. Formative feedback should be non-evaluative, supportive, timely and context specific [12]. Effective tutors use a scaffolding approach after diagnosing student difficulties [7]. Such an approach though highly effective cannot be used in large cohorts with fixed budgets. We posit, pedagogical agents can help fill this gap by augmenting domain knowledge with scaffolding skills of effective tutors.

Design patterns used for modeling complex interaction behaviors in the industry, rely on a good understanding of sequence diagrams (SDs) [5]. However, SDs posed the most difficulties among novices learning modeling [13]. Similarly when we analyze our own modeling tasks in the final exam, we found many students had no idea how SDs were constrained by other models. Many exhibited difficulties in identifying valid interacting-objects and constructing messages with appropriate arguments. Though students understood the role of objects, messages and arguments individually, they were daunted when considering all constraints imposed by other models, concurrently.

The cognitive load theory postulates that the cognitive load resulting from a task may potentially hamper learning [15]. Any strategy that involves more cognitive load than available working memory can deteriorate performance by overwhelming the learner [14]. Modelling SD overwhelms many learners as it involves a high number of interacting items that must be handled concurrently [14]. The direct correlation that exists between cognitive load and self-efficacy [17], helps to explain why students exhibit poor self-efficacy in modelling SDs. We report the results of our ongoing studies where we have gradually raised types of constraints and goals the agent can handle with commensurate levels of support. The main contribution in this paper is to demonstrate how pedagogical agents augmenting domain knowledge with scaffolding techniques can assist novices learning modelling tasks by reducing the cognitive load. Our main research question is:

- Can pedagogical agents augmenting domain knowledge with scaffolding improve the learning outcomes of stragglers modeling sequence diagrams?

2 Related Work

Pedagogical agents are defined to be autonomous agents that support human learning, by interacting with students in the context of an interactive learning environment as a guide, critic, coach or wizard [4]. Pedagogical agents using scaffolding have been shown to enable significant improvement in students' learning outcomes [6]. Scaffolding is timely support given to students to foster problem-solving and design skills [2]. The key features of scaffolding are ongoing diagnosis, adaptivity and fading, but these features are neglected by some developing pedagogical agents for complex environments, often equating scaffolding to additional support [9]. Good tutors are usually able to adjust to the learning style of the student and use a scaffolded approach by giving just enough support to help students solve their problem. However, with increasing class sizes and diversity, tutors cannot provide the levels of support needed [8]. Intelligent agents can be made to give the right amount of hints by tracking the current and goal states and capturing the proficiency level of the learner [7].

3 Overview and Elements of the Pedagogy Agent

Our pedagogy agent permits a scaffolding approach providing gradual feedback on consistency, message validity, completeness and quality. Inputs to the pedagogy agent includes the description of class diagram, methods that must be called specifying particular order if needed, and the quality related metrics. The class diagram supplied together with AI rule-based techniques capturing domain constraints help enforce consistency. For example, the agent forbids a message to be dispatched to a target object if the class it belongs to does not have a corresponding method. The data and methods explicitly capturing data dependencies allow knowledge state in entities to be maintained, preventing data to be dispatched prematurely. In the second stage when student submits the sequence diagram, student will be asked to re-attempt if the specified methods based on use cases are not called or if they are called in incorrect order. In the final stage when a student has submitted a valid sequence diagram, design will be graded based on the qualitative metrics supplied. For example, if distributed design is specified as a quality-criteria, a poor grade will be awarded if most of the messages are originated by the same entity. Figure 1 shows a sample sequence diagram for a class diagram with 4 classes Doctor, Hospital, Patient and Appointment.

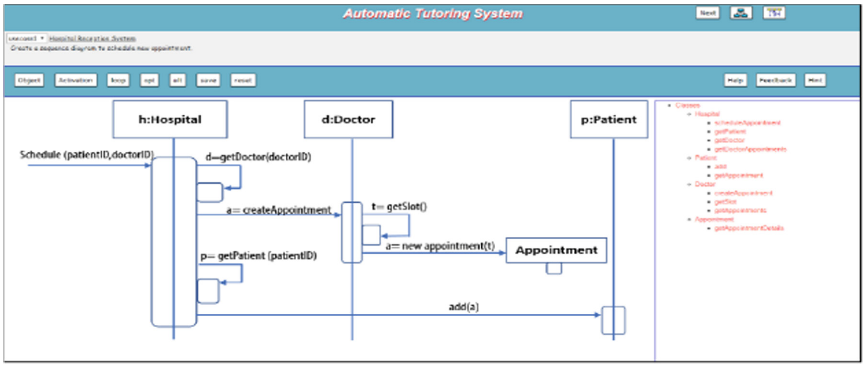

Fig. 1. A sample of a completed Sequence Diagram using our agent which must discharge its responsibilities by calling the Appointment constructor and the *add* method of *p:Patient*.

4 Results

All 243 students taking the first software engineering course were invited to the trial the agent, and out of the 94 students who volunteered only 68 proceeded to complete both tests and the survey. The average marks for pre- and post-tests were 46.25 and 61.25 respectively showing a 32% improvement. To study how the agent affects students with different grades, we analyzed the distribution of pre- and post- test marks in cycle 1, which had a substantial number of students as shown in Fig. 2. The distribution of test marks before and after using the agent suggests weaker students (especially those scoring only 0–49 in the pretests) had the greatest gains. Note the number of students scoring in the range 0–49 declined by nearly 60% from 32 students to 13 students, suggesting a pedagogical agent can significantly improve the performance of stragglers in design activities.

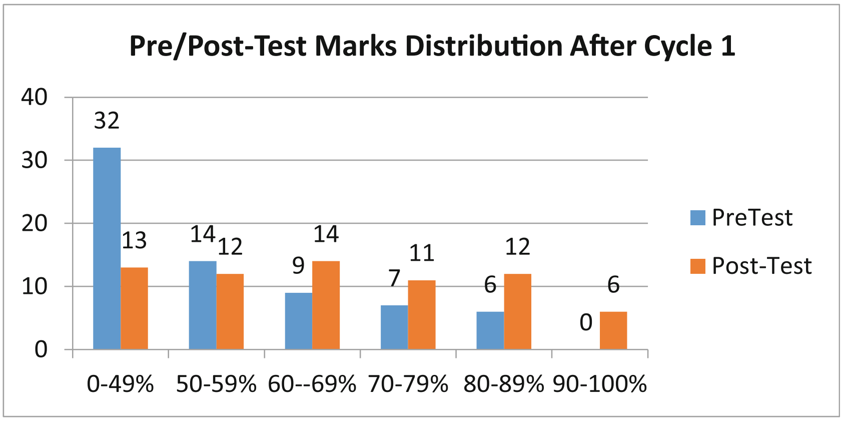

Fig. 2. Marks distribution in pre/posttests after cycle 1

We designed a survey to study the effectiveness of the pedagogical agent from students' perspectives. The survey included Likert-scale and open-ended questions. Students were asked to complete the survey at the end of the modelling activity and tests. The survey was completed by 68 students. Questions were primarily about the agent, student's difficulties and whether the agent can help them get over misconceptions. The results showed around 61.4% of students found learning UML diagrams difficult while over 79% of the students found the modelling agent with instant feedback beneficial for learning UML design. Most of the students found the agent allowed them to grasp the interdependencies between class diagrams and SDs. The overwhelmingly positive response (over 80% agree and over 45% strongly agree) to the three questions related to student confidence, understanding and awareness suggests pedagogical agents can play a key role in improving the self-efficacy of students.

5 Discussion

Success in modeling is generally recognized as requiring a certain level of cognitive development [3]. The cognitive load theory postulates the cognitive load resulting from a task may potentially hamper learning [15]. Decomposing an inherently difficult subject matter can help reduce the cognitive load by allowing subtasks to be first learnt individually [11]. Scaffolding has proven to be effective with diverse student cohorts as it helps to decompose complex problems into incremental constructivist steps [1, 16]. Our solution using a pedagogy agent approach allows cognitive load for modeling SDs to be gradually increased using scaffolding. In the initial stage consistency rules and data dependencies were enforced, before introducing valid completion criteria and grade for quality. Figure 2 depicts most of the weaker students had better learning improvement and displayed greater satisfaction in the second stage where greater scaffolding and multiple tasks were provided accompanied with context specific feedback.

6 Conclusion

Modelling sequence diagrams poses heavy cognitive load on students as constraints and rules imposed by other models must be analyzed concurrently, making it the most poorly performing UML artifact. Effective tutors use scaffolding techniques to teach cognitively demanding tasks. Augmenting a goal and constraint driven agent with such scaffolding techniques appears to substantially improve the learning outcomes in modelling sequence diagrams. The scaffolding techniques allow creation of student specific pathways with varying levels of cognitive challenges and support. The varying levels of support are provided through prompting, feedback, guidance and problem decomposition. Problem decomposition allows cognitive load to be reduced when necessary by enabling students to focus solely on one aspect at a time. This longitudinal study allowed data collected from experienced tutors, lecturers and participants to evolve a more personalized approach to teaching to our increasingly diverse student cohorts.

References

1. Alhazmi, S., Hamilton, M., Thevathayan, C.: CS for all: catering to diversity of master's students through assignment choices. In: Proceedings of the 49th ACM Technical Symposium on Computer Science Education, pp. 38–43, February 2018. https://doi.org/10.1145/3159450.3159464
2. Belland, B.R.: Scaffolding: Definition, Current Debates, and Future Directions. Springer, New York (2014). https://doi.org/10.1007/978-1-4614-3185-5_39
3. Eckerdal, A., Ratcliffe, M., Mccartney, R., Zander, C.: Can graduating students design software systems? ACM SIGCSE Bull. **38**(1), 403–407 (2006). https://doi.org/10.1145/1124706.1121468
4. Jondahl, S., Morch, A.: Simulating pedagogical agents in a virtual learning environment. In: CSCL 2002, Boulder (2002). https://doi.org/10.3115/1658616.1658705
5. Lu, L., Kim, D.: Required behavior of sequence diagrams: semantics and conformance. ACM Trans. Softw. Eng. Methodol. **23**(2), 15:1–15:28 (2014). https://doi.org/10.1145/2523108
6. Martha, A.S.D., Santoso, H.B., Junus, K., Suhartanto, H.: A scaffolding design for pedagogical agents within the higher-education context. In: International Conference on Education Technology and Computers. ACM, Amsterdam (2019). https://doi.org/10.1145/3369255.3369267
7. Merrill, D.C., Reiser, B.J., Ranney, M., Trafton, J.G.: Effective tutoring techniques: a comparison of human tutors and intelligent tutoring systems. J. Learn. Sci. **2**(3), 277–305 (1992). https://doi.org/10.1207/s15327809jls0203_2
8. Parvez, S.M., Blank, G.: Individualizing Tutoring with Learning Style Based Feedback. Intelligent Tutoring Systems, Montreal (2008)
9. Puntambekar, S., Hubscher, R.: Tools for scaffolding students in a complex learning environment: what have we gained and what have we missed? Educ. Psychol. **40**(1), 1–12 (2005). https://doi.org/10.1207/s15326985ep4001_1
10. Schaffer, H.E., Young, K.R., Ligon, E.W., Chapman, D.D.: Automating individualized formative feedback in large classes based on a direct concept graph. Front. Psychol. **8**, 260 (2017). https://doi.org/10.3389/fpsyg.2017.00260
11. Shibli, D., West, R.: Cognitive load theory and its application in the classroom. J. Chart. Coll. Teach. **8** (2019)
12. Shute, V.J.: Focus on formative feedback. Rev. Educ. Res. **78**(1), 153–189 (2008). https://doi.org/10.3102/0034654307313795
13. Sien, V.Y.: An investigation of difficulties experienced by students developing unified modelling language class and sequence diagrams. Comput. Sci. Educ. **21**(4), 317–342 (2011). https://doi.org/10.1080/08993408.2011.630127
14. Sin, T.: Improving Novice Analyst Performance in Modelling the Sequence Diagram in Systems Analysis: A Cognitive Complexity Approach. Florida International University, Miami (2009)
15. Sweller, J.: Cognitive load during problem solving: effects on learning. Cogn. Sci. **12**(2), 257–285 (1988). https://doi.org/10.1207/s15516709cog1202_4
16. Thevathayan, C., Hamilton, M.: Supporting diverse novice programming cohorts through flexible and incremental visual constructivist pathways. In: ACM Conference on Innovation and Technology in Computer Science Education, pp. 296–301. ACM, Lithuania (2015). https://doi.org/10.1145/2729094.2742609
17. Vasile, C., MariaMarhan, A., Singer, F.M., Stoicescu, D.: Academic self-efficacy and cognitive load in students. Soc. Behav. Sci. **12**, 478–482 (2011). https://doi.org/10.1016/j.sbspro.2011.02.059