Learning Environments for Fostering Disciplinary Practices in CS Undergraduates

Deepti Reddy[†], Kavya Alse[*], Lakshmi T. G.[*], Prajish Prasad[*], Sridhar Iyer[*]
[†] SIES Graduate School of Technology, Mumbai, India
[*] Indian Institute of Technology Bombay, Mumbai, India
deepti.reddy@siesgst.ac.in,{kavyaalse,tglakshmi,prajish.prasad,sri}@iitb.ac.in

ABSTRACT

Disciplinary practices are processes and skills applied for sense-making, reasoning and problem solving. Studies have shown that experts are able to spontaneously apply these skills but novices have difficulty due to various reasons. The aim of this paper is to showcase learning environments that foster disciplinary practices in undergraduate computing students in three contexts - data structures, software design and computer networks. Findings from our studies show that students are able to apply these disciplinary practices to solve ill-structured problems. These findings provide motivation for exploring disciplinary practices in other CS courses as well.

CCS CONCEPTS

• **Social and professional topics → Computing education**.

KEYWORDS

Disciplinary practices, TELEs, Learning Environments, Design Evaluation, Design Problem Formulation, Troubleshooting

1 INTRODUCTION

Teaching-learning of computing involves teaching students the necessary skills and practices required in the workplace. However, graduating students encounter ill-structured and complex problems in the workplace. To solve such problems, students need knowledge of certain disciplinary practices. Disciplinary practices are processes and skills applied for sense-making, reasoning and problem-solving [3]. These practices are used by engineers as they design and analyze models and systems [1]. Research has shown that such practices have to be explicitly taught to students, along with necessary content knowledge [2]. In this paper, we showcase learning environments targeted at fostering disciplinary practices in students for courses such as data structures, software design and computer networks.

Woodstock '18, June 03–05, 2018, Woodstock, NY
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

2 BACKGROUND

In this paper, we have focused on the following practices and the corresponding courses in which they are taught - Problem formulation and design (data structures), Integrated model building (software design), Evaluation of a given design (software design), and troubleshooting a network (computer networks). We have chosen these courses as they correspond to our domains of teaching and expertise. Studies have shown that novices have difficulties in these practices [4, 5]. Moreover, there exists a gap in teaching-learning environments which foster these practices. For example, current teaching-learning practices in software design are limited to software methodologies and tools. When troubleshooting is taught in the context of networking, it is usually about the features of the tool rather than tool-independent structured process of troubleshooting.

3 LEARNING ENVIRONMENTS FOR DISCIPLINARY PRACTICES

We have developed technology enhanced learning environments (TELEs) to train students in applying disciplinary practices in specific courses. The underlying pedagogical principle in each of the TELEs is to enable learners to practice the targeted practices by doing, reflecting and monitoring their progress, through solving real-life ill-structured problems. The TELEs have various features such as - question prompts to direct students' thinking in applying the targeted skill, using various visual representations which aid in simplifying complex problems, solved examples, simulations of system behavior, pedagogical agents for hints and feedback, and tasks to promote reflection.

4 RESULTS AND CONTRIBUTIONS

We conducted qualitative and quantitative studies with students in order to investigate the effectiveness of our TELEs. Results from these studies show that students improved in their use of disciplinary practices, and in their approach towards solving ill-structured problems. We plan to conduct further studies, as well as explore disciplinary practices relevant in other CS courses as well.

REFERENCES

[1] Rodger W Bybee. 2014. NGSS and the next generation of science teachers. *Journal of science teacher education* 25, 2 (2014), 211–221.
[2] David H Jonassen. 2004. *Learning to solve problems: An instructional design guide.* Vol. 6. John Wiley & Sons.
[3] Matthew Lipman. 2003. *Thinking in education.* Cambridge University Press.
[4] Alma Schaafstal, Jan Maarten Schraagen, and Marcel Van Berl. 2000. Cognitive task analysis and innovation of training: The case of structured troubleshooting. *Human factors* 42, 1 (2000), 75–86.
[5] Antony Tang, Aldeida Aleti, Janet Burge, and Hans van Vliet. 2010. What makes software design effective? *Design Studies* 31, 6 (2010), 614–640.