e-PRACTICE ENVIRONMENT TO LEARN PROGRAMMING FOR PROBLEM SOLVING COURSE

Rajeev Sukumaran, Vairavel G

Centre for Applied Research in Education, SRM Institute of Science and Technology, India

ABSTRACT

Covid-19 has taught us how to work, study, meet and attend events remotely. This has helped us to be creative, collaborative and find more inner strengths. As academicians, this pandemic has brought out the best in all of us to search, find and utilize all available web based learning tools. All of us have evaluated many software applications, their utility and suitability in higher technical education than ever before. It is in these troubled times, that we have become more innovative in using the TPACK (Technology, Pedagogy and Content Knowledge) curricula. In the last couple of months of intense work, we have analyzed, designed, developed, tested and evaluated an in-house built e-Practice environment. This e-Practice tool that we have built is to learn the “Problem Solving” course available for all Undergraduate first year students in our University. This course is unique and crucial that the learners have to logically and thoroughly analyze the given problem scenarios, then design, implement and execute solutions using computer programming languages. Problem sets posted are real-world scenarios presented in an increasing order of solving complexity. The main objective behind this course is to develop logical thinking abilities of higher order and latent thinking for problem solving, a much needed skill for all graduates. The other objective is also to develop programming skills. Hence, designing and developing an instructor-led e-Practice environment for this course, needed us to substantially understand pedagogy, andragogy, cybergogy and ergonagy, moreover it also demanded us to understand how web-based responsive applications needs to be built using just in time technologies. In this paper, the key aspects of this learning and research experience is shared for wider dissemination of adapting best practices of CDIO implementation.

KEYWORDS

Problem Solving, Online Programming, Integrated Learning, ePractice, Cybergogy, Competitive Coding, Standards: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

INTRODUCTION

The Problem Solving course using programming offered for undergraduate first year students in our University is a fundamental skill course aimed towards developing logical and higher order thinking skills of analysis, justification and evaluation. In the CDIO framework, it helps learners acquire D, I and O. As our University is spread across India with campuses (i) Kattankulathur, (ii) Ramapuram, (iii) Vadapalani around Chennai in TamilNadu, (iv) Sonepat in Haryana, (v) Modinagar in Delhi NCR and (vi) Amaravathi in Andhra Pradesh, we get a wide diverse set of students numbering more than 20,000 in the first year who undergo the course.

In this Problem Solving Course, learners are uniquely provided with a set of real-time problem scenarios ranging from simple to complex situations to analyze. Learners analyze the scenarios provided and find solutions and code the same using computer languages. Even though there is no specific programming language preference, however students tend to solve mostly using C, C++, Java or Python languages (which they have learnt in school education) using laboratory computing facility loaded with their required compilers and IDE's as required. The arrived solutions are further submitted to their assigned facilitators for review and assessments to identify how they are able to solve the given case study.

This Pandemic has forced us all to facilitate learning using online modes only, and hence the order of the day was to use any available simulation tools or virtualization tools to learn and practice. But simulators are not real and virtualization is also not real. Hence, there arouse a need to build a robust, reliable web-based remote access e-Practice tool that could handle the large capacity of students who could access and solve problems posted on their respective logins. Moreover, there are no such open source or free software currently available for this e-Practice for problem solving.

The structure of this paper is organized as follows: (a) Requirement Analysis (b) Design and Development of e-Practice environment, (c) Learning Process using e-Practice tool (d) Benefits of using e-Practice Tool and (e) Conclusion and Future Work

REQUIREMENT ANALYSIS

To build a remote access e-Practice entity, we identified the following requirements:

(i) Should support D,I and O of the CDIO framework.

(ii) Learners should be able to use any internet connected digital devices that they currently possess such as phone, tab, laptop, desktop, net connected TVs etc.,

(iii) Solving problems can be using any web-browser in any operating system.

(iv) There shall not be any application/software to load into their devices, rather they could access the entire e-Practice tool via a simple web-browser. (Ala-Mutka, 2004)

(v) Should not consume much of the 3G/4G data bandwidth that would tax the limited connectivity available to rural students.

(vi) Should support our existing Technology Pedagogy and Content Knowledge (TPACK) curricula framework. (Shin et al., 2009)

(vii) Should have support to compile C, C++, Java, Scilab, Octave, Python, Haskell, Lua, Swift, Rust, Julia, Java Script, R, Ruby, C#, Perl, Clojure, Go, Shell Script and SQL programming languages.

(viii) The e-Practice environment should not be a Simulator or Virtual Entity, rather a remote access real-time environment. (Hiltz & Turoff, 2005)

(ix) Problem Solving questions shall invoke one or more of the logical thinking abilities such as Inductive, Deductive, Abductive, Defeasible, Statistical, Probabilistic and Reflective Reasoning. (Tiruneh et al., 2015)

(x) Problem Solving questions shall provoke exploration, imagination, and creativity. (Koedinger & Aleven, 2007).

(xi) The posted solutions for the Problems shall be auto-evaluated, (Thiebaut, 2015) thus reducing the strain on teachers who otherwise have to physically go through every possible solution put forth.

(xii) The e-Practice tool shall support: student login and interactions, faculty login with batches assignments, co-ordinator login with problem entry and report generations.

(xiii) The e-Practice environment shall be developed only using Free and Open Source technologies. (Hsieh et al., 2013).
(xiv) Tool should support Peer-based learning, Active-Collaborative Learning, Mobile based programming Strategies, Problem Based Learning, and Case Based Learning.
(xv) In order to provide the assistance to students based on their needs and knowledge level, the problem sets should be structured from simple to complex.

DESIGN AND DEVELOPMENT OF e-PRACTICE ENVIRONMENT

The e-Practice environment's main focus is on determining how people learn (the science of learning) and how to help people in learning (the science of instruction) in problem solving practices. The tool is designed to be built as a Client-Server model. With the server hosting all the required program compilers and the front end client with responsive interface screens.

Some of the design features include:
(i) Session wise learning outcomes well defined, and corresponding problem sets created.
(ii) For each learning session, the problem sets are categorized into the levels of thinking following Blooms taxonomy (Buckley & Exton, 2003).
(iii) More number of problem sets are created, so that learners get random real-time scenarios; thereby avoiding copying of solutions from one another.

The design architecture for developing the e-Practice tool is given below:

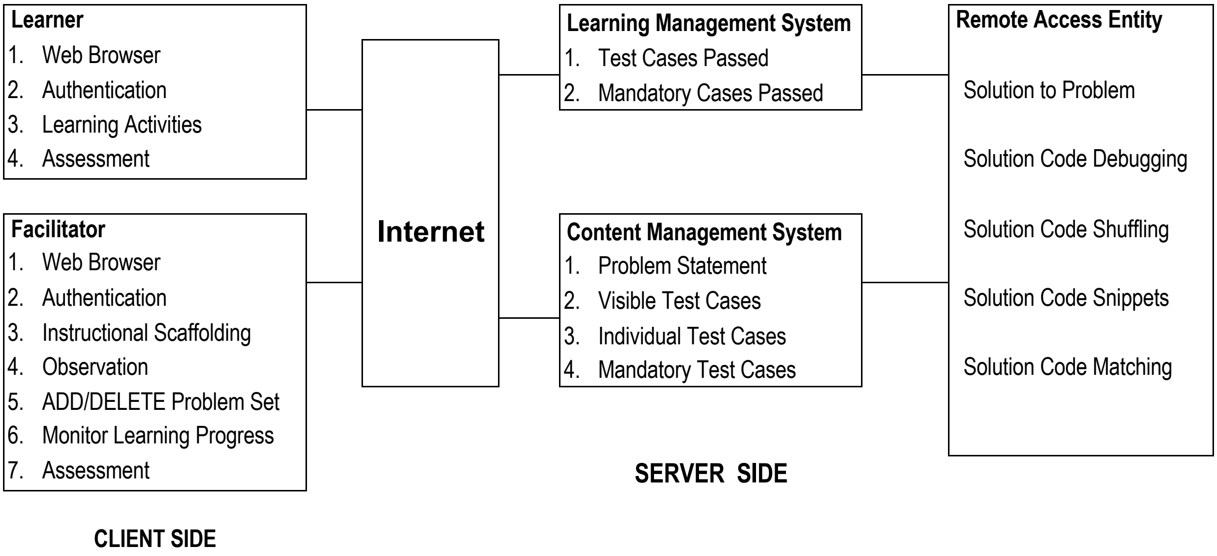


Figure 1. e-Practice Tool : Client Server Model

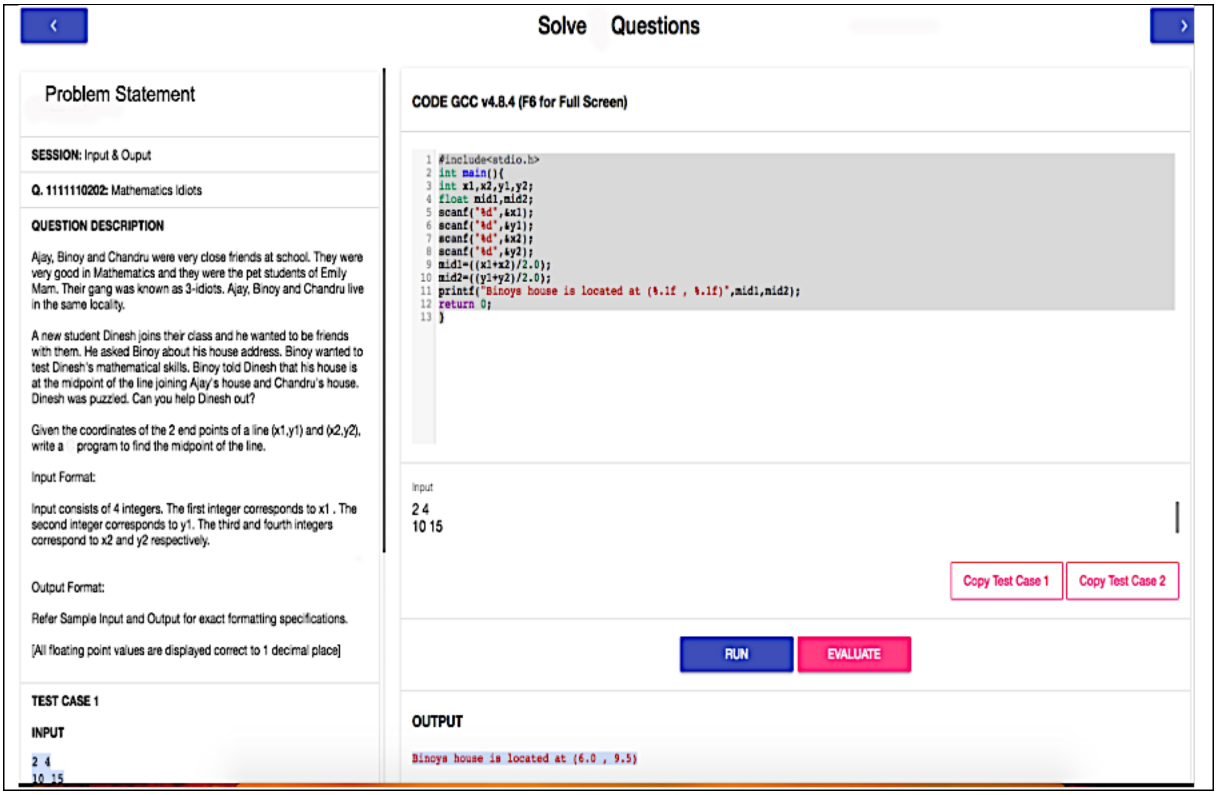
The e-Practice Environment client/server model is represented in the Figure 1. The Server side modules include the Content Management System (CMS), Learning Management System (LMS) and the Remote Access Programming Environment. The CMS contains the problem statements and the test cases for program correctness verification such as; visible test cases, invisible test cases and mandatory test cases. The LMS contains the solution program evaluation module and it provides five learning activities such as; (i) Solution to problem

(ii) Solution code debugging, (iii) Solution code shuffling, (iv) Solution code snippets and (v) Solution code matching.

The course facilitator can define and create new problem sets, and for each problem set, they can define the problem statement, its test cases (visible, invisible and mandatory conditions). Test cases are a set of correct solutions to the problem statement, to help determine the accuracy of the solution identified by the learner. Any good educator must know the students' needs and their current cognitive levels to guide them in a better way during the learning process.

LEARNING PROCESS USING e-PRACTICE TOOL

For the learners, a simple user interface for "Problem Statement" is provided to see the problem descriptions. A sample problem description is shown in Figure 2.



Problem Statement

SESSION: Input & Ouput

Q. 1111110202: Mathematics Idiots

QUESTION DESCRIPTION

Ajay, Binoy and Chandru were very close friends at school. They were very good in Mathematics and they were the pet students of Emily Mam. Their gang was known as 3-idiots. Ajay, Binoy and Chandru live in the same locality.

A new student Dinesh joins their class and he wanted to be friends with them. He asked Binoy about his house address. Binoy wanted to test Dinesh's mathematical skills. Binoy told Dinesh that his house is at the midpoint of the line joining Ajay's house and Chandru's house. Dinesh was puzzled. Can you help Dinesh out?

Given the coordinates of the 2 end points of a line (x1,y1) and (x2,y2), write a program to find the midpoint of the line.

Input Format:

Input consists of 4 integers. The first integer corresponds to x1 . The second integer corresponds to y1. The third and fourth integers correspond to x2 and y2 respectively.

Output Format:

Refer Sample Input and Output for exact formatting specifications.

[All floating point values are displayed correct to 1 decimal place]

TEST CASE 1

INPUT

```
2 4
10 15
```

CODE GCC v4.8.4 (F6 for Full Screen)

```
#include<stdio.h>
int main(){
int x1,x2,y1,y2;
float mid1,mid2;
scanf("%d",&x1);
scanf("%d",&y1);
scanf("%d",&x2);
scanf("%d",&y2);
mid1=((x1+x2)/2.0);
mid2=((y1+y2)/2.0);
printf("Binoys house is located at (%.1f , %.1f)",mid1,mid2);
return 0;
}
```

Input

```
2 4
10 15
```

OUTPUT

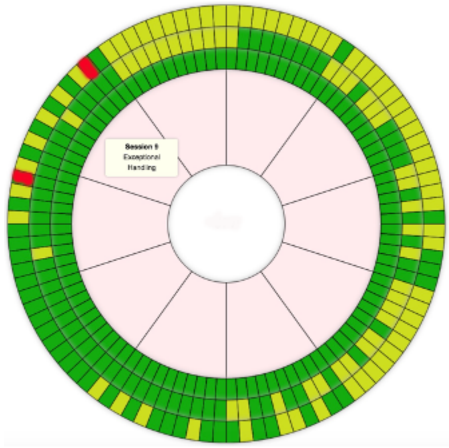
```
Binoys house is located at (6.0 , 9.5)
```

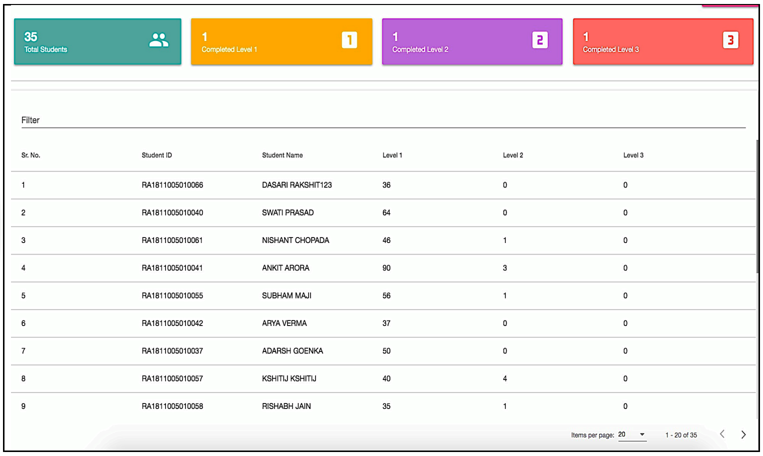
Figure 2. e-Practice Tool : Learner User Interface

The "Editor" workspace is used to program the solutions for the provided problem statement. Learners can solve the given problem using any of the programming languages of choice like: C, C++, Java, Scilab, Octave, Python, Haskell, Lua, Swift, Rust, Julia, Java Script, R, Ruby, C#, Perl, Clojure, Go, Shell Script or SQL.

The solutions are compiled real-time in the server and the status of the solution is updated duly to the respective course facilitator, who can cross-check anytime the status of any learner registered.

Every learner registered with the e-Practice tool are provided with a total of 300 problem statements from a pool of 2500+ problem statement repository. These 300 problems are divided into 100 problem statements each of 3 levels (easy, medium and hard).




35 Total Students

1 Completed Level 1

1 Completed Level 2

1 Completed Level 3

Sr. No.	Student ID	Student Name	Level 1	Level 2	Level 3
1	RA1811005010066	DASARI RAKSHIT123	36	0	0
2	RA1811005010040	SWATI PRASAD	64	0	0
3	RA1811005010061	NISHANT CHOPADA	46	1	0
4	RA1811005010041	ANKIT ARORA	90	3	0
5	RA1811005010055	SUBHAM MAJI	56	1	0
6	RA1811005010042	ARYA VERMA	37	0	0
7	RA1811005010037	ADARSH GOENKA	50	0	0
8	RA1811005010057	KSHITIJ KSHITIJ	40	4	0
9	RA1811005010058	RISHABH JAIN	35	1	0

Items per page: 20 1 - 20 of 35

Figure 3. Problem Status

Figure 4. Learning Progress

Figure 3. shows 10 sessions, each with 10 problem statements per level and there are 3 levels. The status colors of Red representing learner not attempted, yellow represents that learner is trying to solve the given problem and green representing the learner has successfully found a solution and the e-Practice tool has auto-evaluated the solution.

The course facilitator can see the progress of each learner and their levels of completion (Figure 4), this helps the facilitator to identify learners who are struggling and the ones who are smartly solving. Hence, facilitators can invest their time and efforts in mentoring and helping the academically challenged learners to develop their logical reasoning abilities and programming skills. Figure 5 shows the overall learning progress of a batch of learners.

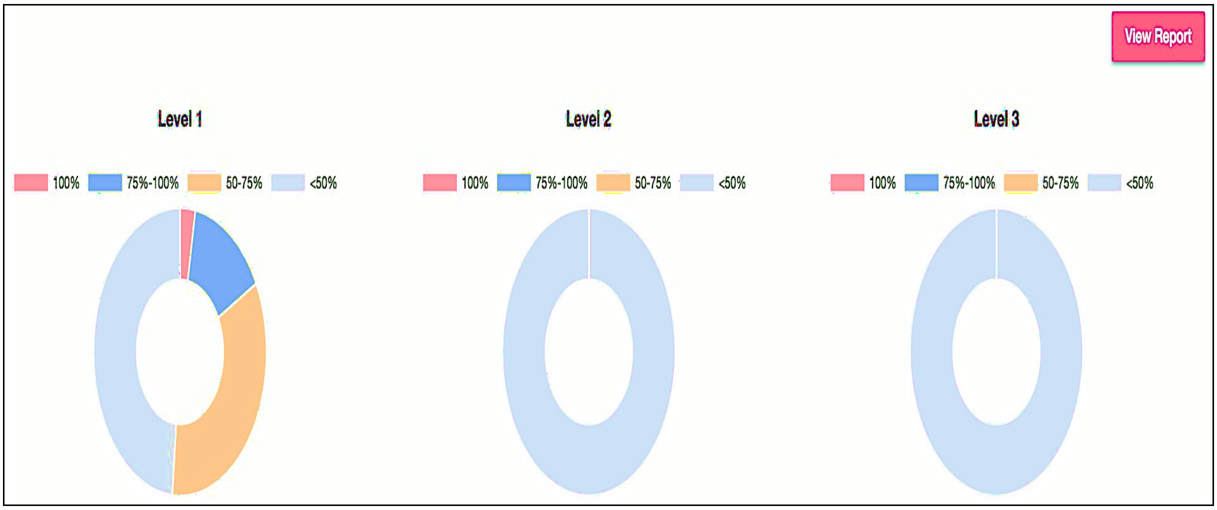


Figure 5. Overall Learning Progress

Documentation for each level of learning achievement is prepared at the end of each session. It is a tool designed to detect plagiarism, whereby submitted solutions are also scanned against a large database of previously submitted solutions.

BENEFITS OF USING e-PRACTICE TOOL

The benefits obtained by using the e-Practice tool for the course “Programming for Problem Solving” are given under:

Learner Benefits

(i) An easy to use browser based e-Practice platform to enhance problem solving skills
(ii) Helps learner to reflect on their learning
(iii) Identify learners based on their skills and learning abilities
(iv) Analytics and dashboards that indicate the top areas of interests
(v) Motivates through digital reflection
(vi) Stimulate problem solving skills and life-long learning skills.
(vii) Increase awareness about ‘how they are learning’.
(viii) Increase the responsibility for their learning.
(ix) Enables learners to engage in a deeper rather than surface knowledge.
(x) New information is acquired through self-directed learning.

Course Facilitators Instructional Benefits

(i) Supports CDIO and TPACK framework
(ii) Provides Activity Based Learning environment
(iii) Help learners practice cognitively complex tasks
(iv) Assist learners to examine errors in reasoning
(v) Help learners revise knowledge
(vi) Notice learners involvement and engagement
(vii) Provide feedback and mentoring
(viii) Use of plagiarism-detection services to report on content
(ix) Identify learner performance (Fast Learners, Slow Learners and Poor Learners)
(x) Facilitates learning measurement, continuous assessment and evaluation

CONCLUSION

The main purpose of developing and utilizing the e-Practice tool was to help students to continue their learning curve despite all odds created by the present pandemic. If we as teachers can do a little bit of additional effort to build a platform that supports learning from their homes for a course that is most sought after in campus, then it is worth every bit of it. This e-Practice environment was deployed for the learners of our University, who have registered for study under the regular, distance education and online learning modes. During this pandemic times, we have reached out to more than 20,000 students for continuing their learning despite all regulatory restrictions. Personally as creators of this entity, we feel immensely happy to see learning continue as it is supposed to be.

REFERENCES

Ala-Mutka, K. (2004). Problems in learning and teaching programming–a literature study for developing visualizations in the Codewitz-Minerva project. *Codewitz Needs Analysis*, Literature Study 1-13.

Buckley, J., & Exton, C. (2003, May 10-11). Bloom's taxonomy: a framework for assessing programmers' knowledge of software systems. In Program Comprehension [Paper presentation]. *11th IEEE International Workshop on Program Comprehension*, Portland, OR, USA. doi: 10.1109/WPC.2003.1199200.

GyorÃ, C., GyorÃ, R., & Sotoc, R. (2015). A Comparative Study of Relational and Non-Relational Database Models in a Web-Based Application. *International Journal of Advanced Computer Science & Applications*, 6(11), 78-83. http://dx.doi.org/10.14569/IJACSA.2015.061111

Hiltz, S. R., & Turoff, M. (2005). Education goes digital: The evolution of online learning and the revolution in higher education. *Communications of the ACM*, 48(10), 59-64. https://doi.org/10.1145/1089107.1089139

Hsieh, T. C., Lee, M. C., & Su, C. Y. (2013). Designing and implementing a personalized remedial learning system for enhancing the programming learning. Journal of *Educational Technology & Society*, 16(4), 32-46. http://www.jstor.org/stable/jeductechsoci.16.4.32

Koedinger, K. R., & Aleven, V. (2007). Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, 19(3), 239-264. https://doi.org/10.1007/s10648-007-9049-0

Pamuk, S., Ergun, M., Cakir, R., Yilmaz, H. B., & Ayas, C. (2015). Exploring relationships among TPACK components and development of the TPACK instrument. *Education and Information Technologies*, 20(2), 241-263. https://doi.org/10.1007/s10639-013-9278-4

Paraiso, F., Challita, S., Al-Dhuraibi, Y., & Merle, P. (2016, June 27 – July 02). Model-driven management of docker containers [Paper Presentation]. *IEEE 9th International Conference on Cloud Computing* (CLOUD), San Francisco, CA, USA. https://ieeexplore.ieee.org/document/7820337

Shin, T. S., Koehler, M. J., Mishra, P., Schmidt, D., Baran, E., & Thompson, A. (2009, March 02). Changing technological pedagogical content knowledge (TPACK) through course experiences [Paper presentation]. *Society for information technology & teacher education international conference*, Charleston, SC, USA. https://www.learntechlib.org/p/31309/

SRM-CARE. (2016). elabstarter. https://care.srmist.edu.in/#/elabstarter

Thiebaut, D. (2015). Automatic Evaluation of Computer Programs Using Moodle’s Virtual Programming Lab (VPL) Plug-in. Journal of Computing Sciences in Colleges, 30(6), 145-151. https://dl.acm.org/doi/10.5555/2753024.2753053

Tiruneh, D. T., Weldeslassie, A. G., Kassa, A., Tefera, Z., Cock, M., & Elen, J. (2015). Systematic design of a learning environment for domain-specific and domain-general critical thinking skills. *Educational Technology Research and Development*, 64, 481-505. https://doi.org/10.1007/s11423-015-9417-2

BIOGRAPHICAL INFORMATION

Rajeev Sukumaran is a Senior Advisor at the SRM Centre for Applied Research in Education (SRM-CARE), SRM Institute of Science and Technology (University). His expertise includes engineering epistemology, education, educational psychology and philosophy. He is instrumental in setting up teaching learning centres across various higher technical institutions in India. He is also a Senior IEEE member and a Fellow of Institution of Engineers.

Vairavel G is currently a Professor in the SRM Centre for Applied Research in Education (SRM-CARE). His research interests are Engineering Education, Instructional Technology, CDIO curricula design and implementations. He is also a Senior IEEE member.

Corresponding author

Rajeev Sukumaran
Centre for Applied Research in Education
SRM Institute of Science and Technology
Kattankulathur, Chengalpattu, TamilNadu
India 603203
advisor.care@srmist.edu.in