ASEE's 123rd Annual • Conference & Exposition • New Orleans, LA • June 26-29, 2016

New Orleans JAZZED about Engineering Education

Paper ID #14997

Applying ”The New Age of Innovations Principles” to Software Engineering Education

Dr. Pradeep Kashinath Waychal, Guruji Education Foudation

Dr Pradeep Waychal is a founder trustee and the chair of Guruji Education Foundation that provides holistic support to the education of underprivileged students and operates on funding from friends. The foundation has recently extended its work in diverse areas such research in engineering education, youth employability and teaching computer science to adolescents. Earlier, Dr Waychal has worked at Patni Computer Systems for 20 years in various positions including the head of innovations, NMIMS as the director Shirpur campus and at College of Engineering Pune (COEP) as the founder head of the innovation Center.

Dr Waychal earned his Ph D in the area of developing Innovation Competencies in Information System Organizations from IIT Bombay and M Tech in Control Engineering from IIT Delhi. He has presented keynote / invited talks in many high profile international conferences and has published papers in peer-reviewed journals. He / his teams have won awards in Engineering Education, Innovation, Six Sigma, and Knowledge Management at international events. Recently, his paper won the Best Teaching Strategies Paper award at the most respected international conference in the area of engineering education - Annual conference of American Society of Engineering Education (ASEE). His current research interests are engineering education, software engineering, and developing innovative entrepreneurs and intrapreneurs.

Applying “The New Age of Innovations Principles” to Software Engineering Education

Introduction

The ever-increasing ubiquity and criticality of software requires a mature software engineering discipline. However, it is still an evolving and young discipline, [1, 2, 3] which is causing changes in the character of software development [4]. Educating students in such a discipline presents difficulties but also offers opportunities for innovations in teaching and developing transferrable skills. We leveraged these opportunities by using the principles from the New Age Of Innovation of Prahalad and Krishnan[5].

Software engineering education researchers and practitioners have made significant innovations [6,7,4] but they do not seem to have explored the application of new age innovation principles [5]. Of the two principles - (a) N=1 (every customer is unique with unique requirements) and (b) R=G (global resources can help in handling the plurality of the requirements) - we are focusing on the former, i.e. N=1, in this paper.

The evolving nature of software engineering, as compared to many other engineering disciplines, offers some benefits in using the principle. The principle implies that an organization must focus on individual customers (or stakeholders), even if it is serving millions of them. As an example, a health insurance company mandates its customers to regularly check their health, collects the medical data online, and adjusts premium. The customers who follow a healthy lifestyle receive the benefit of a reduced premium. Customers, who do not undergo regular checkups, end up paying a higher premium. This encourages them to undergo regular medical tests and take proper care of their health. Similarly, we can apply the principle in software engineering education. We can collect relevant information and based on that provide optimized value to each student. Proving that N=1 principle can be successfully applied is a major contribution of the paper.

The next section describes our study method that includes data analysis, interpretation, and conclusion.

Method of Study

We have showed the method of study in Figure 1. It defines the objective and scope, and explains the selected instruments to develop better understanding of individual students and comments on reliability and validity of the instruments. The method then describes the sample chosen for our study and dwells on activities deployed to add value to individual students. Data analysis, interpretation, and conclusion are the last elements of the method.

Objective and Scope

The objective of the study was to ascertain if principle N=1 can be successfully applied to a software engineering course and to assess the value it delivers to students. Towards that, we have chosen a set of measurements to develop relevant understanding of the students and planned appropriate activities to add optimized value to each of them. We have come across use of individual measurements and activities but not their integration in this fashion. We chose a software engineering course taught to junior students of a computer-engineering department at one of the premier colleges in XXX for our study.

Instruments and their Reliability and Validity

Learning individuals follow different learning styles and approaches. Hawk and Shah [8] assert that the proposition that students learn and study in different ways has emerged as a prominent pedagogical issue within the last three decades. While we have to plan classroom and laboratory sessions by considering all approaches, we can customize one-on-one discussions and self-study sessions to learning approaches and styles of individuals. Students have to be aware that they need to occasionally step out of their preference and work using other styles and approaches. Learning styles and approaches characterize students at study; therefore, they were included in the assessment.

Teamwork (team behavior), which includes egoless behavior, is an important dimension, especially in case of software engineering. The IEEE software engineering body of knowledge [2] states that a software engineer must be able to interact cooperatively and constructively with others to first determine and then meet both needs and expectations. Therefore, we have considered the team behavior (teamwork) to characterize students.

Learning Approach

We chose RASI (Revised Approaches to Studying Inventory) instrument for helping students discover their learning approaches. Entwistle et al. [9] have developed the first version of Approaches to Studying Inventory (ASI). It has undergone many revisions and today researchers use various versions of RASI.

We used a short 30-question instrument that assesses three learning approaches – deep, surface, and strategic. Students following a deep approach look for meaning in what they are learning and enjoy the learning activity. Surface learners primarily use memorization and not logical reasoning to learn. They do not connect to learning from other courses and sources, and are, most likely, in the wrong study program. Students following a strategic approach are determined to excel in academics, have well organized studying habits, and manage time well. They choose deep and surface approaches selectively. Duff [10, 11] has carried out two studies to ascertain the reliability and validity of the 30-question version of the RASI.

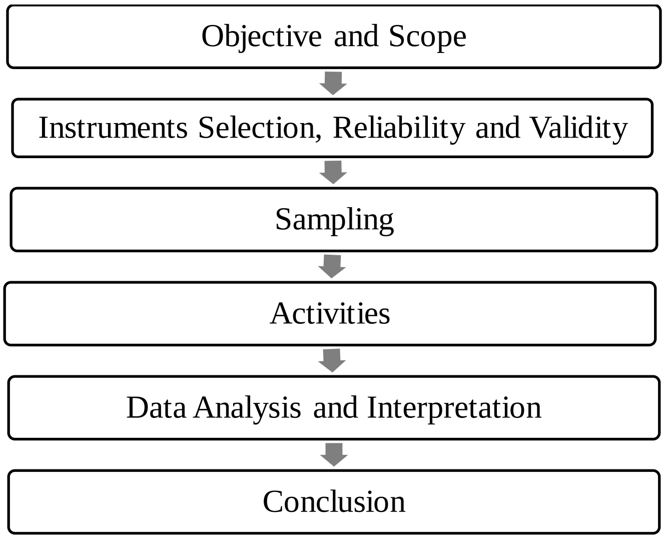
Objective and Scope
Instruments Selection, Reliability and Validity
Sampling
Activities
Data Analysis and Interpretation
Conclusion

Figure 1: Study Process

Learning Style

We relied on the Index of Learning Style (ILS) that assesses preferences on four learning style dimensions using a model developed by Felder and Silverman[12]. The model defines learning style as 'the characteristic strengths and preferences in the ways individuals take in and process information' and asserts that individuals have preferences along four bipolar dimensions: Active-Reflective, Sensing-Intuitive, Visual-Verbal, and Sequential-Global. Hawk and Shah have described the styles as follow [8]. Active learners prefer doing things, particularly in groups. Reflective learners work better alone with time to think about the task before doing it. Sensing learners like facts, data, and experimentation and work well with detail. Intuitive learners prefer ideas and theories, particularly when they can grasp new ideas. Verbal learners like to hear the information and engage in discussion, especially when they can speak and hear their own words. Visual learners like words, pictures, symbols, flow charts, diagrams, and reading books. Sequential learners prefer linear reasoning, step-by-step procedures, and material that comes to them in a steady stream. Global learners are strong integrators and synthesizers, making intuitive discoveries and connections to see the overall system or pattern. Both innate personality traits and prior experiences can affect student preferences on each of those scales. This instrument is originated in the engineering discipline, is very well documented and supported, and is freely available for academic institutes. Felder and Spurlin[13] have cited various studies that have proven the reliability and validity of the ILS.

Teamwork

We used two different instruments for assesing teamwork of students – Layton's instrument[14] and an egoless instrument[15]. Layton's instrument focuses on behavioral characteristics of good teamwork and has good reliability[14]. Weinberg[16] has put forth the concept of egoless programming many years ago. Lamont Adams elaborated the concept with ten commandments. Waychal[17] used these ten commandments to develop an instrument to measure egoless behavior of students, which is important for teamwork[15]. He has found the instrument to be reliable[17].

Personal Information

We sought information on students' academic performance (Cumulative Grade Point Average (CGPA) and Semester Grade Point Average (SGPA)), the subjects that they liked and disliked and their performances in them and their volitional activities such as academic electives , co-curricular and extra-curricular activities. They also furnished details of their socio-economic backgrounds and their career aspirations.

Course outcome

We used course-end feedback to measure the course outcome. The feedback sought additional information about course elements that improved students' learning experience and two things that they learned in the course.

Sampling

Our sample consisted of 88 undergraduate junior students of the computer-engineering program at a respectable college. While most of the students were admitted to the four-year undergraduate (UG) engineering program after 12 years schooling, a few of them (12) had lateral entries in the second year of the program after ten years of schooling followed by three years of an engineering diploma. The college is among the best in the state. It attracts brighter students but they have noticeable variation in performances at the entrance examinations and in the prior courses of the engineering program. Their social and geographical background also had significant variety.

Activities

We designed and executed the following activities in line with N=1 principle. We tried to understand the uniqueness of students through multiple activities such as learning styles, approaches, and team behavior assessments. We observed and peer-evaluated individuals through debating sessions and discussion forum participation. Their academics, social profile, and career plans also helped us to know them better.

Based on all these inputs, we planned different presentations for different individuals and conducted one-on-one meetings to discuss their career plans and course progress. Everyone had to make a presentation; seventy out of the eighty-eight students had one-on-one meetings.

Learning approach and learning style assessment

We assesed the learning approaches and styles of each student at the start of the semester. We explained the theory and applications of the instruments. We emphasized that the assessments were not course tests with right or wrong answers. We requested the students to provide their natural responses, if they wanted to benefit from the assessments.

Teamwork

We deployed project- and team-based learning (PBL-TBL) strategy for the course. Early in the semester, we assigned students to think of a real-life software application that they could develop. We evaluated their ideas and randomly formed teams of 5-6 students. In the first team meeting, all members presented their application ideas and debated to choose their projects. They chose one of the presented ideas, developed a hybrid of some of the ideas, or picked an entirely new idea. The student teams discussed the project ideas with the course faculty who moderated the size and scope of the projects. After a few weeks, when students had spent sufficient time with the projects and team members, we administered Layton's and the egoless questionnaires. We explained the importance of teamwork in their professional lives and that the assessments, while not a part of the course grades, were important improvement mechanisms. The students assessed themselves and their team members resulting in self-assessment and peer-validation reports, which we shared with the respective students. We posited that becoming aware of self-assessment and their peer-validation could help in bringing about the required improvement.

Debates

We asked all students to read and study seminal papers such as 'No Silver Bullets' [18], 'Cathedral and Bazaar' [19], and 'On the Criteria to be Used in Decomposing Systems into Modules' [20]. We organized debates on learning from the papers and facilitated peer evaluations. We had another assignment in which students browsed the Internet, watched around five videos in software engineering, and chose the video that they liked the most. They had to announce the video they have chosen on the Moodle discussion forum and ensure that nobody else had taken it before them. In case someone else had, the individual had to choose another video. We organized the students into groups of 10 and instructed the groups to screen and discuss the videos led by the individual who had chosen the video. The groups rated each member on value addition of the video, ability to lead the discussion, and participation in the discussion of other videos. A faculty member observed the discussions and moderated the ratings.

Discussion forum

We used Moodle to administer a discussion forum. We tracked the quantitative and qualitative contributions of each student. We used that information in one-to-one discussion as well as for grading. The faculty and students chose thought-provoking topics, such as 'Software testing cost and benefits tradeoffs', 'Is software engineering, science, art, or craft?' and 'Agile Methods and Process Discipline'.

Academic, social profile, and career plans

We sought academic and social profiles that helped us to understand students and their career aspirations.

Presentations

We leveraged all the acquired information about students to customize their assignments. Students who were aspiring to get into management careers worked on presentation assignments such as 'Data driven marketing' and 'Are ERP (Enterprise Resource Planning) implementations successful?' The ones who were planning to get into administration services were steered to topics of social importance such as 'Using software to solve energy problems' and 'Implementation of a stock exchange system'. Those who were research-oriented had opportunities to study and speak on topics such as 'Random number generations' and 'Knuth-Morris-Pratt Algorithm'. The faculty worked closely with the students when they were developing their presentations. One of the students was interested in studying law. When a police officer visited the faculty for expert opinion to resolve a dispute between a software developer and his customer, the faculty invited this student to join the discussion and report it to the whole class.

One-on-one discussions

We offered opportunities to students to discuss their plans, in general, and progress in the course, in particular in one-on-one meetings. The results of their assessments, information on academic performance in the course tests and assignments, and their socioeconomic profile helped in having meaningful discussions. The discussions resulted in customized value addition to the students. Over the semester, 70 out of 88 students utilized this offer.

Data Analysis and Interpretation

We asked students to list course elements that provided them better learning experiences. We think that the successful implementation of N=1 principle can be largely verified with the help of students' feedback. They have to feel that we have met their personal educational needs. We have plotted their feedback in Figure 2. The collaborative learning and N=1 principle had the highest rating of 3.8 on the Likert scale of 1 to 5. We succeeded in team activities as well as in paying attention to individuals. Learning approaches, learning style evaluations, grading schemes, and others were at around 3.5. 'Others' included activities such as screening of podcasts, opportunities to develop friendship with team members, use of real life examples, rewarding right answers by giving chocolates, audio visuals and presentations, and study of seminal papers. The exam type scored the least, i.e., around 3.1. We had weekly tests for the course which most of the students disliked. The examinations used to create stress, even though each test had no more than 2 % impact on the final marks.

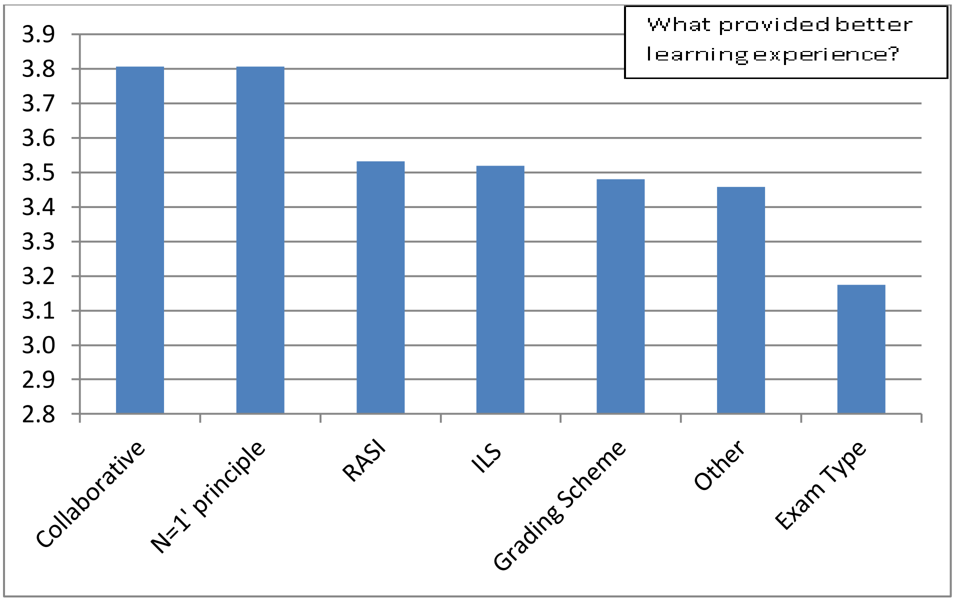

Figure 2: What provided better learning experience? (5-point Likert scale)

We also asked students to list the two most important things that they learned in the course. Out of 170 entries from 85 students, 99 belonged to technical and 71 to non-technical or transferrable skills areas. This means that we could utilize the course in evolving discipline to develop transferrable skills.

Students tend to think that developing software is nothing but coding. Software engineering courses must remove that misconception and make students understand the holistic nature of software engineering. The course succeeded in that, as the highest number of students voted for that learning. The importance of processes and testing followed the next. In non-technical areas, the learning skill was the topmost. Teamwork and the benefits of egoless behavior followed that. We have presented these details in Figures 3a and 3b.

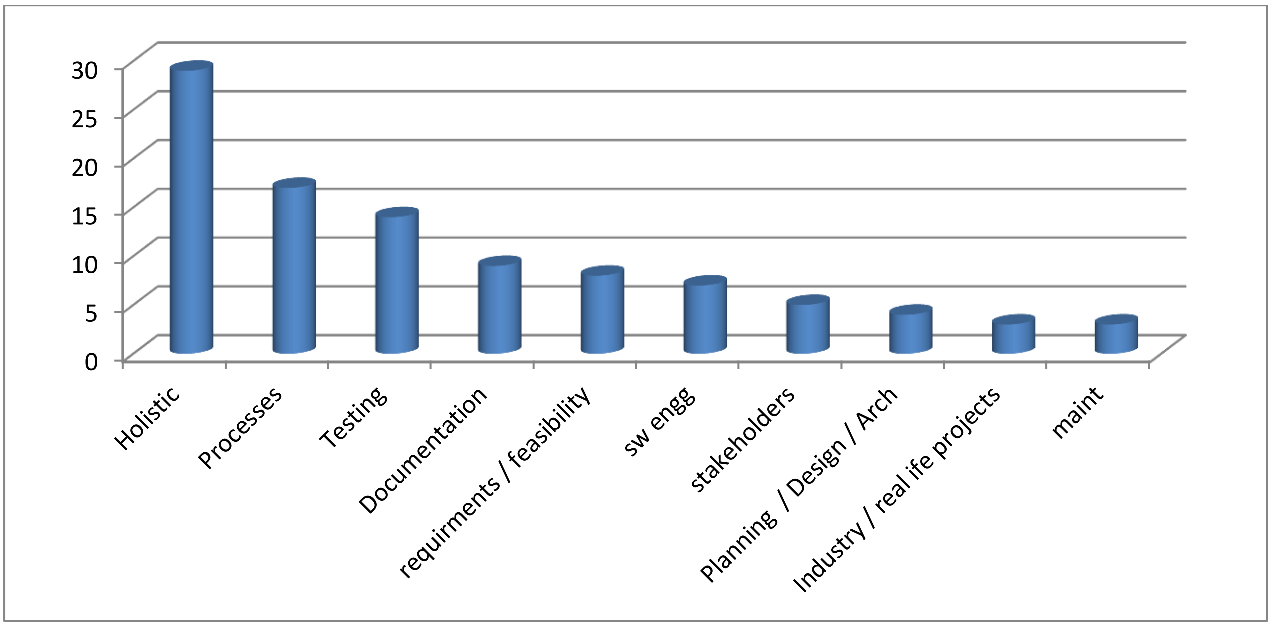

Figure 3a: Learning from technical areas

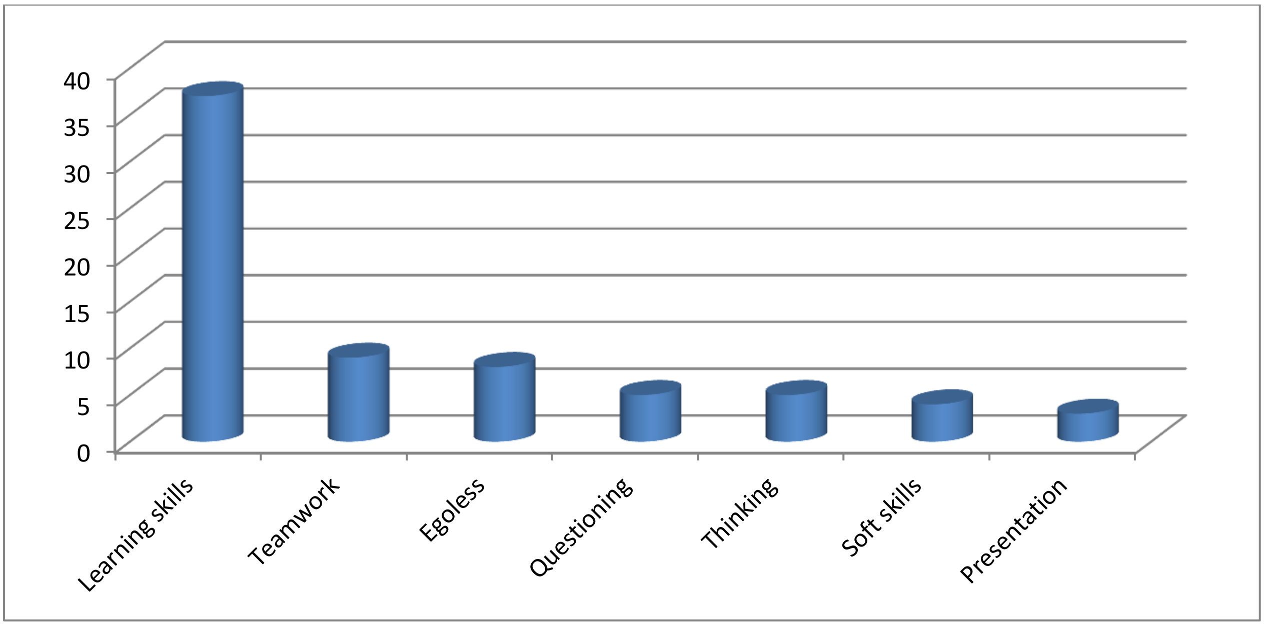

Figure 3b: Learning from non-technical areas

Conclusion

Software engineering is a relatively young and fast-evolving discipline. It is natural that education in such a discipline would require continuous experimentation and innovation. This paper discusses the application of a new age of innovations principle, N=1, to a junior-level course in software engineering that achieved notable success. We treated students as our customers, identified their unique characteristics, and attempted to serve their holistic academic needs. This included assessments of their learning styles, learning approaches and team behavior. It also included understanding students' socio-economic and academic backgrounds consisting of elective and core courses that they liked and their performance in them. We also utilized student-centered learning strategies such as project-based and team-based learning to learn more about individual students.

All the above student information allowed us to implement N=1 principle. At the end of the course, we asked students about the course elements that offered them better learning. The feedback highlighted the impact of the principle. While some researchers have questioned the efficacy of learning style assessments [21-22], some others have found them to be helpful [8- 23]. In our experience, they indeed added value. We also asked students about their learnings and found that they had both technical and non-technical learnings. Their feedback indicates that we could leverage the evolving nature of the discipline and provide enough opportunities for self-learning.

We argue that principle N=1 will help faculty members in adding value to their students in the upcoming MOOC (Massive Online Open Courseware)-dominated world. Some educators, especially those in developing countries, are questioning their roles in the light of the MOOC onslaught. We argue that they can augment usage of the MOOC by customizing various course elements and add significant value to individual students. We need to study the application of the principle beyond geographical and subject (software engineering) boundaries.

The study does have some limitations. We relied on feedback and not performance of students for evaluating results of the study. We think that the successful implementation of N=1 principle can be largely verified with the help of students' feedback. They have to feel that we have met their personal educational needs. That could translate into improved learning outcomes, which we have to explore. Further, we could not use random control design to qualify the study as a research experiment. Cost and availability played significant roles in the choice of the assessment instrument set. We can optimize the set in subsequent iterations of the study. As we do that, we may be able to develop a better platform for implementing the new age innovation principles, which can add significant value to students and faculty.

Acknowldgement

We thank all students who participated in the exercise. We also thank Mr. Abhay Joshi and anonymous reviewers who, with their insightful comments, helped us improve quality of the paper.

References

1 Mary Shaw, 'Prospects for an Engineering Discipline of Software', *Software, IEEE,* 7 (1990), 15-24.

2 IEEE, *Guide to the Software Engineering Body of Knowledge*. ed. by James W. Moore Alain Abran, Pierre Bourque, Robert Dupuis (Los Alamitos, California: IEEE Computer Society, 2004).

3 Bourque Pierre 'The Maturation of Software Engineering as a Discipline and Recognized Profession'2009) <http://sites.ieee.org/southern-alberta/files/2011/12/oct26.pdf>.

4 Mary Shaw, 'Software Engineering Education: A Roadmap', in *Proceedings of the conference on The future of Software Engineering* (ACM, 2000), pp. 371-80.

5 Coimbatore Krishna Prahalad, and Mayuram S Krishnan, *The New Age of Innovation: Driving Cocreated Value through Global Networks*. Vol. 1 (McGraw-Hill New York, 2008).

6 Jesűs Favela, and Feniosky Peńa-Mora, 'An Experience in Collaborative Software Engineering Education', *Software, IEEE*, 18 (2001), 47-53.

7 Jeffrey Carver, Letizia Jaccheri, Sandro Morasca, and Forrest Shull, 'Issues in Using Students in Empirical Studies in Software Engineering Education', in *Software Metrics Symposium, 2003. Proceedings. Ninth International* (IEEE, 2003), pp. 239-49.

8 Thomas F Hawk, and Amit J Shah, 'Using Learning Style Instruments to Enhance Student Learning', *Decision Sciences Journal of Innovative Education*, 5 (2007), 1-19.

9 Noel Entwistle, Maureen Hanley, and Dai Hounsell, 'Identifying Distinctive Approaches to Studying', *Higher education*, 8 (1979), 365-80.

10 Angus Duff, 'A Note on the Reliability and Validity of a 30-Item Version of Entwistle & Tait's Revised Approaches to Studying Inventory', *British Journal of Educational Psychology*, 67 (1997), 529-39.

11 Angus Duff, 'Quality of Learning on an Mba Programme: The Impact of Approaches to Learning on Academic Performance', *Educational Psychology*, 23 (2003), 123-39.

12 Richard M Felder, and Linda K Silverman, 'Learning and Teaching Styles in Engineering Education', *Engineering education*, 78 (1988), 674-81.

13 Richard M Felder, and Joni Spurlin, 'Applications, Reliability and Validity of the Index of Learning Styles', *International journal of engineering education*, 21 (2005), 103-12.

14 Matthew W Ohland, and Richard A Layton, 'Comparing the Reliability of Two Peer Evaluation Instruments', in *2000 ASEE Annual Conference and Exposition Proceedings* (2000).

15 L Adams, 'Ten Commandments of Egoless Programming', *Tech Republic* (2002).

16 Gerald M Weinberg, *The Psychology of Computer Programming*. Vol. 932633420 (Van Nostrand Reinhold New York, 1971).

17 Pradeep Waychal, 'Introducing an Approach to Develop Egoless Software Engineers', in *ASEE Annual Conference* (Seattle: 2015).

18 Fredrick Brooks, 'Essence and Accidents of Software Engineering, ' *IEEE Computer*, 20 (1987), 10-19.

19 Eric Raymond, 'The Cathedral and the Bazaar', *Knowledge, Technology & Policy*, 12 (1999), 23-49.

20 David L Parnas, 'On the Criteria to Be Used in Decomposing Systems into Modules', in *Software Pioneers* (Springer, 2002), pp. 411-27.

21 Harold Pashler, Mark McDaniel, Doug Rohrer, and Robert Bjork, 'Learning Styles Concepts and Evidence', *Psychological science in the public interest*, 9 (2008), 105-19.

22 Frank Coffield, David Moseley, Elaine Hall, and Kathryn Ecclestone, 'Learning Styles and Pedagogy in Post-16 Learning: A Systematic and Critical Review', (2004).

23 Richard M Felder, 'Are Learning Styles Invalid?(Hint: No!)', *On-Course Newsletter* (2010), 1-7.