Assessment of Student's Learning Style And Engagement in Traditional Based Software Engineering Education

N. Pratheesh
Department of Computer Applications
School of Computer Science and Engineering
Bharathiar University, Coimbatore – 641 046, India
pratheesh_n@hotmail.com

T. Devi
Department of Computer Applications
School of Computer Science and Engineering
Bharathiar University, Coimbatore – 641 046, India
tdevi5@gmail.com

***Abstract*–Software Engineering courses are nucleus elements of the Computer Science syllabus. While the main aspire of such courses is to give students practical industry-relevant "software engineering in the large" familiarity, frequently such courses plummet short of this significant intention owed to be short of industrial experience and support infrastructure, degenerating the course into "one big coding assignment". Therefore, it is obligatory to design and develop improved infrastructure support for teaching or taking such courses. This would benefit instructors and students communities in computer science atmosphere. Learning analytics lend a hand to the learners to enhance their learning tricks. This was scrutinized among the software engineering students how the learning style and learning engagement sways in gathering knowledge. This paper discusses the importance of learning analytics in software engineering education especially the learning style, learning engagement and its influences in this domain.**

Keywords–Software Engineering Education, Learning Analytics, Social Learning Analytics, Learning Style, Collaborative Learning

I. INTRODUCTION

Computer software is the single most important technology worldwide. Software has enabled the creation of new technologies such as genetic engineering, the extension of technologies such as telecommunication, and the demise of older technologies such as printing industries. Software engineering education should groom students for industry careers by providing an experience close to real-world practice. However, in most software engineering courses, students develope programs from scratch that, due to time limitations, not have the characteristics of industrial software such as large size, complex architecture, numerous features, strict code quality requirements, and various software artifacts [15]. Graduates fashioned by the existing undergraduate/postgraduate computing educational system have been censure in their willingness to face this new computing environment. Often the new graduates have insufficient experience in open-ended project work and group work [11]. There is a widening gap between the knowledge and experience of our graduates, and the expectations of the software development industries [1]. One of the most effective learning methods in computer science is "learning by example". Well annotated code examples have long-lasting influences on students and computing professionals in their learning, understanding, and practice of fundamental computing principles [18]. Many of the available software packages, though of high quality, are poorly documented and annotated. From this large collection of raw code, it is often difficult for an instructor to identify and organize appropriate materials for software engineering education [11].

The opportunities for informal learning in the modern world are abundant and diverse, and greatly expand on earlier notions like "just-in-time" or "found" learning. Sense-making and the ability to assess the credibility of information are paramount [17]. Mentoring and preparing students for the world in which they will live and work is again at the forefront. Universities have always been seen as the gold standard for educational credentialing, but emerging certification programs from other sources are eroding the value of that mission daily [17]. Students already spend much of their free time on the Internet, learning and exchanging new information often via their social networks. Institutions that embrace face-to face/online hybrid learning models have the potential to leverage the online skills learners have already developed independent of academia [17].

This paper discusses the importance of learning analytics in software engineering education especially the learning style and its influences in this domain. Section II of the study describes the issues in software engineering education. Learning analytics and its importance in software engineering illustrates in section III and section IV focuses on necessity to measure the learning style in software engineering students. Empirical evidences conferred in section V and finally section VI express the conclusion of this study.

II. ISSUE IN SOFTWARE ENGINEERING EDUCATION

Software engineering education plays more significance role in the computer science arena. Learning software engineering is tremendously challenging task, because it consist of massive areas of knowledge such as requirements analysis, software solution architecture, testing, project

978-1-4799-0476-1/13/$31.00 © 2013 IEEE

estimation and project management. Industries are expected that the new recruit is well versed with necessary programming and software engineering knowledge and skills by virtue of the academic training and is prepared to take up professional responsibilities in industry [14]. Software Engineering courses or training programs emphasized on deeper understanding of the topic and highlights on cognitive learning goals of knowledge and understanding. Higher order cognitive ambitions especially application and analysis, followed by evaluation and synthesis, since these skills are highly used and valued by the industry. Most of the training programs follow the traditional approaches of lecture based training and performance based assessments, these approaches are not only less effective, but are also labor intensive on part of training [14].

Software engineering education pattern is divided into modules and each module converse the individual sections and is covered through a series of lectures. Assessments are done at the completion of each module and there may be a final cumulative assessment at the end of the training program [14]. This model of learning never satisfies the industries needs and expectations of the clients and changing market conditions and neglect to impart the constantly incorporating newer technologies, techniques, tools, methods and standards [14]. In general, the students that come to the industry are good in following the syntax, semantics, logic and process (willingly or unwillingly), but are not well in software engineering because usually, software engineering is offered as just one of the subject in a computer science course and most of the computer science graduates study software engineering for at most one semester and for some students this is the only opportunity to get familiar with software engineering before starting their career as Software Engineers [14]. It becomes very important that the course is rich in contents and satisfies at least the basic expectations of the software industry, which is very difficult for a one semester course [14].

Software engineering syllabi are very procedure oriented, and structured around Software Development Life Cycle (SDLC) behaviors. The courses start off with the need of software engineering, followed by SDLC and various phases of SDLC. Very limited time is prescribed for umbrella activities such as quality, estimation, project management etc [22]. Industry and academia considers the skills such as problem solving, self-learning, and professional communication important [22], very few academic courses touch upon these skills, especially problem solving and self-learning [16]. Some Computer Science programs offer a problem solving course clubbed along with a programming course, but do not explicitly teach problem solving techniques or heuristics. Many software engineering curricula include soft skills such as communication, team-building, etc. as a separate course offering, but communication specific to software engineering industrial practice is again rare [20]. Almost all software engineering courses in India use the traditional class-room teaching based instruction approach and this may be useful for imparting theoretical knowledge [14]. There is usually a one to two semester gap between attending the software engineering concepts and applying the learning in the project [14]. This is adequate for students fail to remember the fundamental principles they have learned and the time they need to apply them. Thus, students find it difficult and irrelevant to apply their software engineering learning for such projects, in turn get frustrated and develop the strong notion that software engineering education is a theoretical and is not a useful subject [10, 20, 21].

Principles and theories of software engineering education that may not be directly applicable, and yet students should be motivated to learning them because they shape their mentality and improve their approach to solving practical problems. Because the recognition of the value of principles may come only later, even after years of experience, some "faith and trust" is needed from the students [2]. Good software engineer must have the capabilities such as the theoretical foundations of the discipline, design methods of the discipline, technology and tools of the discipline, keep his/her knowledge current with respect to the new approaches and technologies, interact with other people, understand, model, formalize, analyze a new problem, recognize a recurring problem, and reuse or adapt known solutions, manage a process and to coordinate the work of different people. These qualities can learn and adapt swiftly and is possible only the students acquire self-learning and analytical skills to absorb and espouse newer technologies professionally [14].

III. NEED OF LEARNING ANALYTICS IN SOFTWARE ENGINEERING EDUCATION

Learning analytics is one of the fastest mounting fields of technology enhanced learning (TEL), research that has emerged during the last decade. Growth of this field offered in a broadly sequential structure, demonstrating the increasingly rapid pattern of development as new drivers emerge, new fields are appropriated and new tools developed. Tracing the development of learning analytics over time highlights a gradual shift away from a technological focus towards an educational focus, and the introduction of tools, initiatives and methods that are significant in the field today [19]. Learning analytics consists of 'socialized' approaches, which can readily be applied in social settings. These include content analytics – a broad heading for the variety of automated methods that can be used to examine, index and filter online media assets, with the intention of guiding learners through the ocean of potential resources available to them [23]. These analytics take on a social aspect when they draw upon the tags; ratings and metadata supplied by learners [3]. A second group of socialized analytics focuses on the learning dispositions that can be used to render visible the mixture of experience, motivation and intelligences that influences responses to learning opportunities. Dispositions analytics can be regarded as socialized learning analytics when the emphasis is on the learner in a social setting, engaged in a mentoring or learning relationship [6].

Social Learning Analytics (SLA) are strongly grounded in learning theory and focus attention on elements of learning that are relevant when learning in a participatory

online culture [18]. Approaches to analytics that can be classified in this way include intrinsically the social forms of analytic: social network analytics and discourse analytics [7]. Social learning analytics also includes 'socialised' approaches, which can readily be applied in social settings. These include content analytics – a broad heading for the variety of automated methods that can be used to examine, index and filter online media assets, with the intention of guiding learners through the ocean of potential resources available to them [8, 23]. These analytics take on a social aspect when they draw upon the tags; ratings and metadata supplied by learners [3].

Software engineering education composed of enormous areas of knowledge and every area contains more information. It is very intricate to students and teachers to find the proper information for their needs and the retrieved information may not motivate them to study the software engineering concepts. Learning analytics is a new thought which helps to measure and improve the learning. Learning style and learning engagement influence the acquisition of knowledge, since these are considered as influential factors of learning analytics. This supports to improve and inspire the software engineering education and fabricate knowledgeable software engineer to fulfill the industry needs.

IV. MEASURE THE LEARNING STYLE OF SOFTWARE ENGINEERING STUDENTS

Plenty of information located in traditional and technology based learning in software engineering education domain. The successive of the huge information flow in software engineering education is how the students occupy and gain the knowledge from it. Students struggle to get the proper information from the available resources according to their learning interest. Learning style is an important key role to motivate the learning to the knowledge seekers and it differs from one to another. Students choose the learning materials according to their learning style which motivate them to gain knowledge more than the traditional setup. Usually do not deem individual variances of learners and touch on all learners in the same way in spite of their personal requirements and characteristics. However, the individual student plays an essential role in traditional as well as technology enhanced learning in software engineering education. Each student has special needs and characteristics such as different prior knowledge, cognitive abilities, learning styles, motivation, and so on. These individual differences affect the learning process and the reason why some learners find it easy to learn in a particular course, whereas others find the same course difficult [13].

Learners with a strong preference for a specific learning style may have difficulties in learning if the teaching style does not match with their learning style [9]. From theoretical point of view, conclusion can be drawn that incorporating learning styles of students in the learning environment makes learning easier for them and increases their learning efficiency. On the other hand, learners whose learning styles are not supported by the learning environment may experience problems in the learning process [13]. Number of learning style models exists in literature and it majorly classified into five families which are based on some overarching ideas behind the models, attempting to reflect the views of the main theorists of learning styles [4]. The first family relies on the idea that learning styles and preferences are largely constitutionally based including the modalities: visual, auditory and kinaesthetic. The second family deals with the idea that learning styles reflect deep-seated features of the cognitive structure, including patterns of abilities. A third category refers to learning styles as one component of a relatively stable personality type and fourth family learning styles are seen as flexibly stable learning preferences. The last category moves on from learning styles to learning approaches, strategies, orientations and conceptions of learning [4, 5].

First family of learning style chosen for this research because it relies on the idea that learning styles and preferences are largely constitutionally based including the modalities but other families have its own limitations. Visual learners have a preference for seen or observed things, including pictures, diagrams, demonstrations, displays, handouts, films, flip-chart, etc. Auditory learners have a preference for the transfer of information through listening: to the spoken word, of self or others, of sounds and noises. Kinaesthetic learners have a preference for physical experience - touching, feeling, holding, doing, and practical hands-on experiences [4].

V. RESULT AND DISCUSSION

Data for the study was gathered from structured questionnaire for learning style and self administrated questionnaire for student's learning engagement, conducted among first year MCA students. The structured questionnaire developed by V. Chislett and A. Chapman (2005) was adapted for the study. Forty six questionnaires were distributed to first year MCA students. After the screening, forty two questionnaires were fully completed and useable, yielded a response rate of 91%. SPSS version 17.0 was used to analysis the data. The structured questionnaire used to identifies the type of learners such as visual, auditory and kinaesthetic using thirty questions and self administrated questionnaire used to identify the student engagement in learning analytics with the variable of active participation, emotional engagement, avoidance of text book dominated instruction, reflective thinking, student decision making and problem solving choice, behavioural engagement and relevance in traditional based software engineering class.

A. Analyze the Learner Type

The data collected from the respondents were tabulated and analyzed using appropriate statistical techniques. Table I shows the learning category of the respondents of software engineering students and categorized in to three criterion groups, i.e. visual, auditory and kinaesthetic.

TABLE I. LEARNER TYPE

Type	No. of Respondents	Percentage
Visual	13	31%
Auditory	15	36%
Kinaesthetic	14	33%

Software engineering class students were distributed more or less equally among three categories as 13, 15, 14 in visual, audio, kinaesthetic respectively. This portrays that an academician have to follow an instruction methods that satisfies all learning groups which induce the learning of students in the software engineering class. This depicted through the following graph.

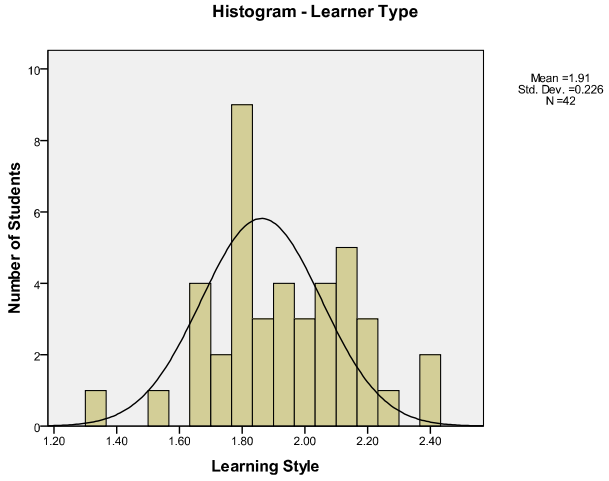

Fig. 1. Distribution of Learner

Fig.1 shows the distribution of learners in the software engineering class with the mean value of 1.91 and standard deviation of 0.23. It shows there is no single learner type of students with in the class room. Software engineering class room occupied with the students of multiple learning styles.

Table II represents the multiplicity of an individual. Most of the software engineering class students do not rely on single learning type and they prefer to adopt multiple learning styles when they study software engineering.

TABLE II. MULTIPLICITY OF AN INDIVIDUAL

Learning Type		Frequency	Percent
Valid	1.33	1	2.4
	1.53	1	2.4
	1.63	4	9.5
	1.73	2	4.8
	1.77	7	16.7
	1.80	2	4.8
	1.87	3	7.1
	1.90	2	4.8
	1.93	2	4.8
	1.97	1	2.4
	2.00	2	4.8
	2.03	2	4.8
	2.07	2	4.8
	2.10	3	7.1
	2.13	2	4.8
	2.17	1	2.4
	2.20	2	4.8
	2.27	1	2.4
	2.37	2	4.8
	Total	42	100.0

However, most of the software engineering class practices traditional teaching method and the way of delivers the knowledge to the students may not satisfy to their learning style. Therefore they feel difficulties in acquiring adequate knowledge in software engineering. It implies that most of the software engineering students' face difficulty to handle the real world problems properly and need not satisfies the industry requirements.

B. Analyze The Level of Student's Engagement

TABLE III. DESCRIPTIVE STATISTIC FOR STUDENT ENGAGEMENT IN LEARNING ANALYTICS

Variables	Mean	Median	Mode	SD
Active Participation	2.2381	2.2000	2.20	0.323
Emotional Engagement	2.6452	2.6000	2.50	0.245
Avoidance of Text book Dominated Instruction	2.6815	2.6875	2.75	0.285
Reflective Thinking	2.5143	2.5000	2.40	0.287
Student Decision Making and Problem Solving Choice	2.1071	2.0625	1.75	0.496
Behavioural Engagement	2.2857	2.2727	2.18	0.237
Relevance	2.3720	2.3650	2.25	0.326

Table III shows the summary of descriptive statistic for first year MCA software engineering students' engagement in learning.

- Active Participation:

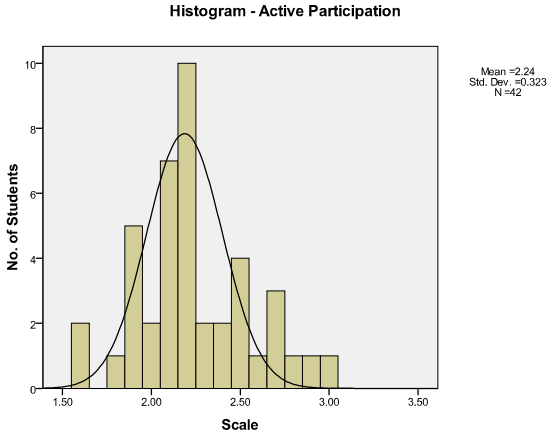

Fig. 2. Distribution of Active Participation

Fig.2 proves that the students distributed with the mean value of 2.24, median of 2.2, mode of 2.2 and standard deviation of 0.323 for the variable active participation and there is no much deviation in opinion among software engineering students. It portrays that the distribution is

skewed positively because the mean value is higher than the median and mode. It shows that traditional based software engineering classes students are not provoked to share their ideas, disagree with views, raise technical issues, and make judgment for the problem. Therefore students are not active in the software engineering traditional based classes.

- Emotional Engagement:

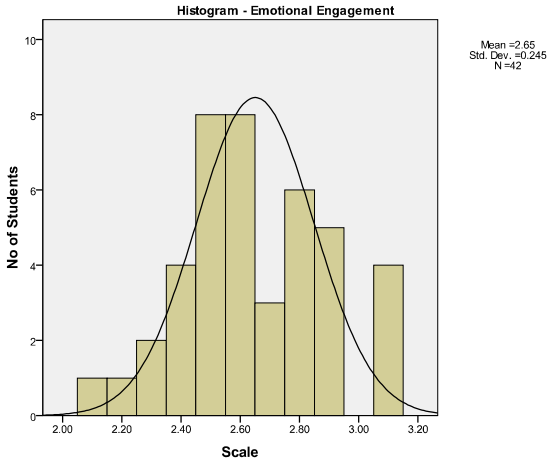

Fig. 3. Distribution of Emotional engagement

Fig.3 confirms that the students distributed with the mean value of 2.64, median of 2.6, mode of 2.5 and standard deviation of 0.245 for the variable emotional engagement and there is no much divergence in attitude among software engineering students. It depicts that the distribution is skewed positively since the mean value is more than the median and mode. It indicates that most of the students prefer the collaborative learning than the pure traditional based system, because they felt that existing teaching method is boredom and do not create much interest in learning process

- Avoidance of Text book Dominated Instruction:

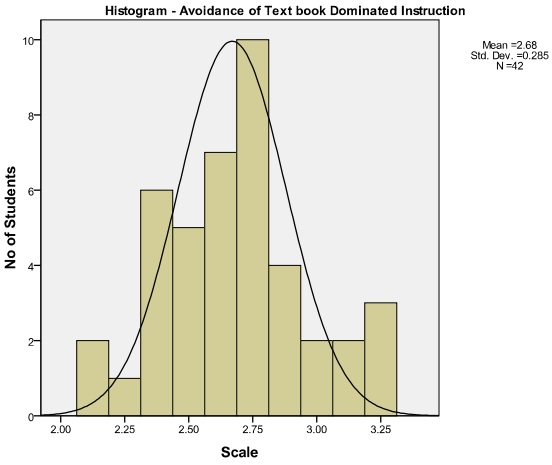

Fig. 4. : Distribution of Avoidance of Text book Dominated Instruction

Fig.4 attests that the students distributed with the mean value of 2.68, median of 2.69, mode of 2.75 and standard deviation of 0.285 for the variable avoidance of text book dominated instruction and there is no much variation in opinion among software engineering students. It describes that the distribution is skewed negatively while the mode value is higher than the median and mean. It exposes that most of the students choose to update their knowledge and find the solution for software engineering problems from watching software engineering lecture videos, browsing information from internet, reading magazines, newsletters, books and articles, chatting with colleagues, discussion forum, etc, than the traditional dictation method of teaching.

- Reflective Thinking:

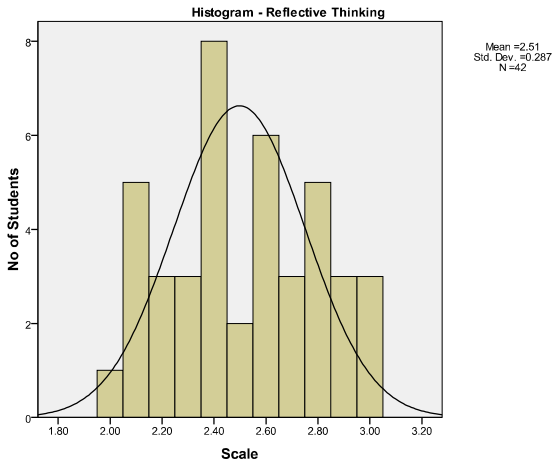

Fig. 5. Distribution of Reflective Thinking

Fig.5 bear outs that the students distributed with the mean value of 2.51, median of 2.5, mode of 2.4 and standard deviation of 0.287 for the variable reflective thinking and there is no much difference in attitude among software engineering students. It renders that the distribution is skewed positively because the mean value is exceeding the median and mode. It reveals that most of the software engineering students expect encouragement from teachers to counter different opinions and gathering knowledge through discussion forum rather than focus on facts and memorization. Further they prefer to have the discussion in practical issues and the use of technology in software engineering concepts.

- Student Decision Making and Problem Solving Choice:

Fig.6 provide evidences that the students distributed with the mean value of 2.11, median of 2.06, mode of 1.75 and standard deviation of 0.496 for the variable student decision making and problem solving choice and there is no much deviation in opinion among software engineering students. It renders that the distribution is skewed positively while the mean value is higher than the median and mode. It divulges that majority of the students perceive they do not have enough freedom to participate and influence the decision making in the traditional based software engineering classes.

Further they perceive they don't have abundant opportunity to make up their own minds about issues related to software engineering concepts, teachers determine the class activities and force them to enroll the activities.

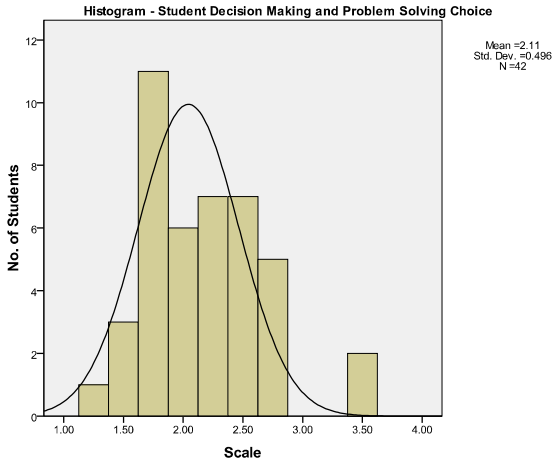

Fig. 6. Distribution of Student Decision Making and Problem Solving Choice

- Behavioural Engagement:

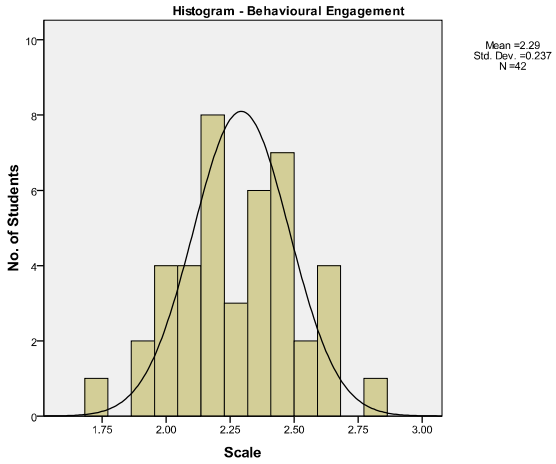

Fig. 7. Distribution of Behavioural Engagement

Fig.7 substantiates that the students distributed with the mean value of 2.29, median of 2.27, mode of 2.18 and standard deviation of 0.237 for the variable behavioural engagement and there is no much digression in opinion among software engineering students. It represents that the distribution is skewed positively since the mean value is higher than the median and mode. It exhibits that most of the software engineering students tend to have disruptive behavior such as skipping lectures and getting in trouble in traditional learning method. This makes students to have low involvement in learning effort, persistence, concentration, attention, asking questions and contribution to class discussions and classroom activities.

- Relevance:

Fig.8 demonstrates that the students distributed with the mean value of 2.37, median of 2.36, mode of 2.25 and standard deviation of 0.326 for the variable relevance and there is no much divergence in opinion among software engineering students. It shows that the distribution is skewed positively while the mean value is exceeding the median and mode. It provide an evidence, software engineering students are of the opinion that technology based course make them more marketable in their chosen field, since it coincide with the challenging real world issues compare to traditional-based learning environment.

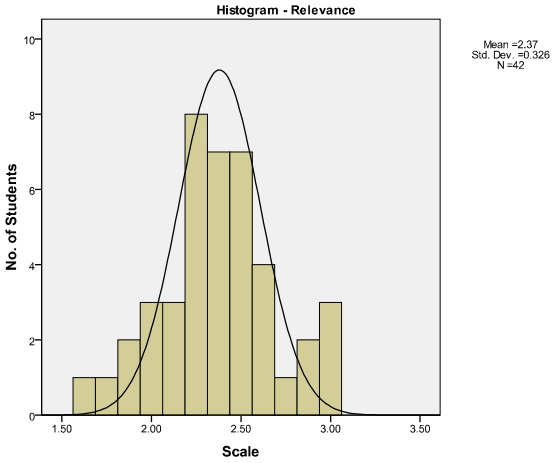

Fig. 8. Distribution of Relevance

VI. CONCLUSION

Software engineering education commonly practices the traditional method teaching. According to the research, a class contains all types of learner categories such as visual, auditory and kinaesthetic, in nearly equal in strength. Here teacher relinquishes the knowledge in class which convinces a certain learner group of students while the mode of teaching create a center of attention a group of learners and this type of learning students learn agreeably than the other learner groups of students. To overcome this issue, teacher has to think about the type of learners in the software engineering class and alter little in his/her teaching style which have room for all types of learner group. According to the learning engagement, majority of the software engineering students prefer collaborative learning environment than traditional based and they feel that this motivates them to study software engineering in depth and would overcome the issue of lack of knowledgeable software engineer to the industries.

REFERENCES

[1] ACM/IEEE Joint Task Force on Computing Curricula. Computing Curricula: 2005 Overview Reportwww.acm.org/education/curricula.html.

[2] Carlo Ghezzi, Dino Mandrioli, "The Challenges of Software Engineering Education", ICSE 05, St. Louis, Missouri, USA. ACM, May 15-21, 2005, pp. 637-638.

[3] Clow. D., & Makriyannis. E. "iSpot Analysed: Participatory Learning and Reputation", Paper presented at LAK11: 1st International Conference on Learning Analytics and Knowledge, Banff, Canada, 27 February – 1 March, 2011.

[4] Coffield, F., Moseley, D., Hall, E., and Ecclestone, K, "Should We Be Using Learning Styles? What Research Has to Say to Practice", Learning and Skills Research Centre / University of Newcastle upon Tyne, London, 2004.

[5] Coffield, F., Moseley, D., Hall, E., and Ecclestone, K, "Learning Styles and Pedagogy in Post-16 Learning: A Systematic and Critical Review", Learning and Skills Research Centre/University of Newcastle upon Tyne, London, 2004.

[6] Deakin Crick. R., Broadfoot. P., & Claxton. G., "Developing an effective lifelong learning inventory: the ELLI project", Assessment in Education: Principles, Policy & Practice, 11(3), 2004, pp. 247-272.

[7] De Liddo, A., Buckingham Shum, S., Quinto, I., Bachler, M., & Cannavacciuolo, L, "Discourse – Centric Learning Analytics", Paper presented at LAK11: 1st International Conference on Learning Analytics and Knowledge, 27 February – 1 March, 2011.

[8] Drachsler, H., Bogers, T., Vuorikari, R., Verbert, K., Duval, E., Manouselis, N., et al., "Issues and considerations regarding sharable data sets for recommender systems in technology enhanced learning", Procedia Computer, 1, 2010, pp. 2849–2858.

[9] Felder, R. M., and Spurlin, J, "Applications, Reliability and Validity of the Index of Learning Styles", International Journal on Engineering Education, 21 (1), 2005, pp. 103-112.

[10] Garg K., Varma V., "A Study of the Effectiveness of Case Study approach in Software Engineering Education", in proceedings of 20th Conference on Software Engineering Education and Training (CSEET 2007), Dublin, July 2007.

[11] Hong Zhang and Hongjun Su, "A Collaborative System for Software Engineering Education", 31st Annual International Computer Software and Applications Conference(COMPSAC 2007), IEEE Computer Society, 2007.

[12] Johnson. L., Adams. S., & Cummins. M., "The NMC Horizon Report: 2012 Higher Education Edition", Austin, Texas: The New Media Consortium, 2012.

[13] Jonassen, D. H., and Grabowski, B. L, "Handbook of Individual Differences, Learning, and Instruction". Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1993.

[14] Kirti Garg, Vasudeva Varma, "Software Engineering Education in India: Issues and Challenges", IEEE Computer Society, 2008, pp. 110-117.

[15] Maksym Petrenko, Denys Poshyvanyk, Václav Rajlich, and Joseph Buchta "Teaching Software Evolution in Open Source", IEEE Computer Society, November, 2007 pp. 25–31.

[16] Melody M. Moore, "Software Engineering Education", IEEE Software, 2002.

[17] Nazim Rahman and Jon Dron "Challenges and Opportunities for Learning Analytics when formal teaching meets social spaces", LAK'12, Vancouver, BC, Country, ACM, 29 April– May 2, 2012.

[18] Rebecca Ferguson and Simon Buckingham Shum, "Social Learning Analytics: Five Approaches", ACM, 2012.

[19] Rebecca Ferguson and SocialLearn, "The State of Learning Analytics in 2012: A Review and Future Challenges", Technical Report KMI-12-01, Knowledge Media Institute, The Open University, UK, 2012.

[20] Shaw M., "Software Engineering Education: A Roadmap", In A. Finkelstein (Ed.), The Future of Software Engineering, New York, NY: ACM Press, 2000, pp. 371-380.

[21] Shaw M.. "Prospects for an engineering discipline of software". IEEE Software, November 1990, pp. 15-24.

[22] Tockey S. R., "Recommended Skills and Knowledge for Software Engineers", In proceedings of International Conference on 12th Software Engineering Education & training, 1999, pp. 168-176.

[23] Verbert. K., Drachsler. H., Manouselis. N., Wolpers. M., Vuorikari. R., & Duval. E. "Dataset—driven Research for Improving Recommender Systems for Learning", Paper presented at LAK11: 1st International Conference on Learning Analytics and Knowledge, Banff, Canada, 27 February – 1 March, 2011.

[24] W. Aspray, A. F. Mayadas, M. Y. Vardi, and S. H. Zweben, "Educational Response to Offshore Outsourcing", In Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education, Houston, Texas, USA,. SIGCSE '06, ACM Press, New York, NY, 330-331, March 03 - 05, 2006.