Indian and Japanese Software Engineering Students in the ”Egoless Space”

Dr. Pradeep Kashinath Waychal, Pune Innovation Centre

Pradeep Waychal has close to 30 years of experience in renowned business and academic organizations. He has been the founder and head of Innovation Center of College of Engineering Pune. Prior to that, for over 20 years, he has worked with a multinational corporation, Patni Computer Systems where he has played varied roles in delivery, corporate and sales organizations. He has led large international business relationships and incubated Centre of Excellences for business intelligence, process consulting and verification and validation. He has headed the corporate product and technology innovations and quality and delivery innovation departments. Pradeep was on the apex senior management group before proceeding on to pursue his academic, research and social interests. Before Patni, he has worked at IIT Delhi, IIT Bombay, SGGS College of Engineering and Crompton Greaves R & D Electronics in different research and academic positions.

Pradeep Waychal has also published papers in peer reviewed journals, presented keynote / invited talks in many high profile international conferences and I involved in a few copyrights / patents. His teams have won a range of awards in Six Sigma and Knowledge Management at international events. He has been associated with initiatives from NASSCOM, CSI, ISO and ISBSG among others. Pradeep Waychal has completed Ph D in the area of Information Technology and Innovation Management from IIT Bombay. He is credited with one of the fastest Ph D even as compared to full time research scholars. He is M Tech in control engineering from IIT Delhi with CGPA of 10/10. He is a graduate from college of engineering Pune in Electronics and Telecommunication. His current research interests are engineering education, software engineering and innovation management.

Gautam Akiwate, University of California, San Diego

Gautam Akiwate is currently a graduate student at the Department of Computer Science and Engineering at the University of California, San Diego with broad areas of interest. He got his bachelor's degree from the College of Engineering, Pune. While in COEP, Gautam was involved in a lot of activities including a CUBESAT mission. Gautam's current research interests are systems and networking in addition to engineering education.

Ms. Ayano OHSAKI, nnovation Center for Engineering Education, Tottori University

Ayano OHSAKI is an assistant professor at the Innovation Center for Engineering Education, Tottori University since 2012. She is in charge of development new engineering education program. The objectives of the program are improvement of creativity, collaboration skills and problem solving skills. Students learn communication skills, project management skills, analysis, etc. by working on design assignments and projects in this program. More than 400 students are studying in this program. She is writing a textbook and developing an assessment system for this program. She is pursuing her doctoral research in Computer Supported Collaboration Leaning (CSCL) and the Flipped Classroom for the Engineering Design. Prior to this, she also has over 5 years of experiences as a Production Engineer. She designed a jig, production processes and production systems for on-vehicle unit systems.

She received the Master of Engineering degree in Information Technology from Shinshu University, Nagano, Japan, in 2009 and BA in education from Saitama University, Saitama, Japan, in 2006. She also underwent training for teacher of technology education, and studied the Stirling engine in the University.

Indian and Japanese Software Engineering Students in the “Egoless Space”

Abstract

Software organizations compete in a highly globalized world mandating continuous improvement in their performance. They need to look beyond the traditional process and technology dimensions and think through the more critical “people” dimension to achieve real improvement. One of the most important initiatives in the people dimension can be developing egoless engineers as professed by Weinberger. This paper proposes creating self-awareness of egoless behavior using a reliable instrument based on Lamont Adams’ "Ten Commandments of Egoless Programming”. It compares and analyses performance of Indian and Japanese software engineering students in the “egoless space”.

Keywords: software engineering, egoless behavior, Indian and Japanese engineering students

Introduction

A software organization typically operates in three dimensions to increase productivity – process, technology, and people. There is considerable literature on processes and technologies[1, 2] but very little on the people dimension[3]. Glass, et al.[3] have studied 369 papers in 6 leading journals and found that Software Engineering research is fundamentally about technical and computing-focused issues and that it is seldom about behavioral issues. In industry, discussion about the people dimension is generally limited to training for the new processes and technologies[4]. Since the approach has not accrued any perceptible gains in productivity[5], we believe that there is a case for exploring the people dimension deeply and earnestly.

The major contribution of this paper is to put forth an instrument to measure egoless behavior as advocated by Weinberger. In the paper, we are using contemporary terms like egoless engineering and development or general terms like egoless behavior to mean the same thing i.e. egoless programming. We have compared the behavior of students from two different countries – Japanese and Indian in the ‘egoless space’.

The paper discusses the background behind the problem of productivity in software organizations followed by the research design leading to analysis of the results and ending with concluding remarks.

Background

In 1993, Potts claimed that ‘all the real problems are people problems’[6]- a statement since supported by researchers and industry pundits alike. Scacchi, in his review of large software engineering projects, found that productivity in projects that were poorly managed or poorly organized was significantly lower[7]. In effect, it can be said that poor management can effectively erase the potential productivity improvements that can be expected from the use of improved technologies and processes[8]. Viljan in his recent work has linked inadequate internal communication and lack of teamwork to a company's weak performance[9]. On the other hand, productive work conditions can be maintained if developers are strongly committed to a project and to team effort[8, 10, 11]. While there is strong and considerable influence of the people dimension on performance and productivity in software organizations, there has been a somewhat weaker agenda from both researchers and practitioners to devise and implement right solutions.

One solution that focuses on the people dimension is “Egoless Programming”. The idea was proposed by Weinberg more than four decades ago[12] wherein he asserted that code quality can improve substantially if personal factors are minimized as the ego of a programmer can get in the way of the effectiveness of review as well as of the sharing and seeking of ideas. Further, egoless development promotes teamwork which is instrumental for success in the real world[4]. The methods and ideas based on co-operation are similar to the ones adopted by the world of free and open source software (FOSS). Eric Raymond has argued that the ‘bazaar’ model - the one adopted in the FOSS world - produces better quality code than the ‘cathedral’ model - the one prevalent in the most software companies[13]. The successful ‘bazaar’ model closely resembles the ‘egoless programming’ and that proves effectiveness of the ‘egoless programming’.

Research Design

This section presents the scope, instrument selection, data collection, reliability assessment, and data analysis.

Scope

Egoless behavior is a mindset. Earlier it is developed, better it is. Younger minds are more malleable and have whole careers ahead of them. Carver,et al. argue that before running an empirical study at a software company, it is useful to carry out a pilot study with students in an academic setting[14]. Therefore, we limited our scope of research to a set of engineering students who had some experience in developing software.

Selection of Instruments

Although egoless programming - as a concept - has been around for nearly four decades; there was no agreement on its objective definition. It was only in 2002, when Lamont Adams put forth 10 factors calling them "Ten Commandments of Egoless Programming” that a step was taken in this direction [15]. These factors, as given in Table 1, seem to have found wide conceptual acceptance. We decided to make use of them to get a measure of "egoless behavior".

Any approach to develop egoless software engineers inherently implies the development of egoless behavior. Hence, the factors would ideally fall into two categories – first the generic factors that correlate to an egoless individual and second coding factors that would correlate to an egoless developer. Considering this, the factors in Table 1 after a group discussion amongst experts were split in the two sets a) Coding Related and b) Generic. The former set consists of the factors C2, C4 and C10 and the latter consists of the rest.

TABLE 1: TEN COMMANDMENTS OF EGOLESS PROGRAMMING

	Ten Commandments of Egoless Programming
C1	Understand and accept that you will make mistakes.
C2	You are not your code.
C3	No matter how much "karate" you know, someone else will always know more.
C4	Don't rewrite code without consultation. There's a fine line between "fixing code" and "rewriting code."
C5	Treat people who know less than you with respect, deference, and patience.
C6	The only constant in the world is change.
C7	The only true authority stems from knowledge, not from position.
C8	Fight for what you believe, but gracefully accept defeat.
C9	Don't be "the guy in the room."
C10	Critique code instead of people—be kind to the coder, not to the code.

Data Collection

The target, of study, was students who have developed some software applications in teams. We started the exercise with Indian students. We chose a random sample of 40 students who assessed themselves on each factor on the Likert scale of 1 to 10. They were requested to indicate any ambiguity or difficulty experienced while responding to the factors, as well as to offer appropriate suggestions. After ascertaining the usability of the instrument, all 86 respondents completed the assessment. We had to exclude one response that had all the factors rated at level 10, leaving 85 valid responses (N1=85). In case of Japanese students, we carried out the survey over web by providing detailed explanation of various questions in the instrument. We received 25 responses. All of them were valid (N2=25). All the students were informed about the purpose of the exercise and the criticality of being egoless in their careers, and were assured full confidentiality of their inputs.

Reliability Assessment

It is important to conduct a thorough measurement analysis on the survey instrument. It gives assurance that the findings reflect accurate measures and that the results are trustworthy. Test reliability further indicates the extent to which individual differences in scores can be attributed to 'true' differences. We used the most popular measure - Cronbach Alpha for assessing reliability of the collected data. Table 2 shows the Alpha values, calculated using Minitab Version 16, for the data collected for each of the subsets.

TABLE 2: CRONBACH ALPHA VALUE FOR THE INSTRUMENT

Subsets	**Alpha Value**
Indian Students	0.864
Japanese Students	0.783

Since alpha values for both the sets were found to be equal to or greater than 0.70, the instrument was judged to be reliable [16].

Analysis and Interpretation of Data

One way stacked ANOVA (Tukey Method) was used to find out grouping between different factors of egoless behavior with the help of Minitab Version 16 (Table 3 and 4). As per Tukey's method, the factors that do not share a letter are significantly different.

For both Indian and Japanese students, all the coding factors were in the lower brackets as compared to generic factors, which meant that coding is a relatively more egoistic activity for students. Amongst the three coding factors, 'You are not your code' was the lowest and the most real factor for them. The students seem to become possessive about their code which is clearly seen in the analysis. They don't get many opportunities either to rewrite or critique code of other developers. Perhaps, therefore, those factors were not in the lowest bracket. The college environment generally has pretty good camaraderie and does not have industry-like intense competition. That would have pushed all the generic factors to the upper bracket.

While the score range for Indian students was very narrow – from 8.23 to 7.51; for Japanese students it was relatively broader – from 9.16 to 5.12. The Indian students, perhaps, were more conservative as they were doing a credit course (even though they were made aware that the exercise will not have any contribution to the grades). The Japanese students may have done more critical evaluation. It could be due to the cultural difference or just the fact that the exercise was not a part of their credit course. Since we were keener to know the ordering than the absolute value, the range difference between the two cohorts did not pose much of a problem.

TABLE 3: INDIAN ENGINEERING STUDENTS: GROUPING INFORMATION USING TUKEY METHOD (N1=85)

Factor	**Mean**	**Group**
The only true authority stems from knowledge not from position	8.23	A
Treat people who know less than you with respect and patience	8.19	A
No matter how much "karate" you know someone else will always know more	8.16	AB
Fight for what you believe but gracefully accept defeat	8.1	AB
Understand and accept that you will make mistakes	8.01	AB
Critique code instead of people – be kind to the coder not to the code	7.99	AB
Don't be "the guy in the room."	7.78	AB
Don't rewrite code without consultation	7.63	AB
The only constant in the world is change	7.62	AB
You are not your code	7.51	B

TABLE 4: JAPANESE ENGINEERING STUDENTS: GROUPING INFORMATION USING TUKEY METHOD (N2=25)

Factor	**Mean**	**Group**
No matter how much "karate" you know, someone else will always know more	9.16	A
Understand and accept that you will make mistakes	8.04	AB
Treat people who know less than you with respect and patience	7.40	ABC
Don't be "the guy in the room."	7.24	ABCD
Fight for what you believe, but gracefully accept defeat	7.12	BCD
The only true authority stems from knowledge, not from position	6.60	BCDE
Critique code instead of people – be kind to the coder, not to the code	6.20	BCDE
Don't rewrite code without consultation	5.96	CDE
The only constant in the world is change	5.44	DE
You are not your code	5.12	E

Conclusion

Software engineering has become an all pervasive discipline with practically every organization relying on it. While software engineering holds huge promise, it is plagued with problems and is unable to deliver the expected performance in terms of productivity, quality and turnaround times. This challenge, we believe, may require an interdisciplinary solution. We argue that this human intensive branch of engineering needs to move beyond traditional initiatives in processes and technology dimensions and start working at the intersection of human sciences and software engineering. We posit that one of the key things that impacts performance of software projects – as professed by Weinberger - is ‘egoless behavior’ of team members. We have proposed a reliable and self scoring instrument to measure the behavior.

Engineering activities, in general, and software engineering activities, in particular, have scaled up to the global level. Most of the projects tend to have global and cross cultural teams. The people oriented software engineering discipline has to understand dynamics of such teams. This prompted us to study egoless behavior of students from two different cultures.

We administered the egoless instrument on a set of Indian and Japanese students. Both the sets have showed similar behavior in the egoless space. The coding-related behavior is poorer than general behavior in case of both the Indian and Japanese students. The Japanese students have used relatively broader range than the Indian students. While there are many syntactical differences like language, food; there seem to be many semantic similarities like work ethos, family and work values between the two cultures.

The measure can be validated with the help of peer assessment. Further, we can create egoless index of teams based on the self and peer assessments. An appropriate action plan, including changes in the team, can be prepared to improve team indices to assure better performing teams.

We require using the measurements to bring in changes in the behavior of individuals. We also require carrying out more experiments to validate the findings of this first experiment and understand finer aspects of the two cultures. Further, we have to experiment in different settings including geographical areas, various types of software organizations and confirm the findings. We are sure that the work done so far, can establish the foundation for further research. Moreover, the findings can be applied in industrial settings in order to maximize organizational performances.

Acknowledgements

At the outset, we would like to thank all the students who participated in the exercise. We also would like to thank Dr. Niimura who administered the survey to the students in Shinshu University, Japan and Mr. Abhay Joshi who reviewed the early manuscripts of the papers and brought in the linguistic improvements.

References

[1] Broman David, “The Company Approach to Software Engineering Project Courses”, IEEE Transactions of Education, vol. 55, no. 4, p. 445, 2012.
[2] Barthélémy Dagenais, “Moving into a New Software Project Landscape,” in IEEE Conference on Software Engineering, Cape Town, South Africa, 2010.
[3] R.L. Glass, I. Vessey, V. Ramesh, “Research in software engineering: an analysis of the literature”, Information and Software Technology, Volume 44, Issue 8, 1 June 2002, Pages 491-506, ISSN 0950-5849,
[4] N. N. Zowghi D, “A study of the impact of requirements volatility on software project performance,” in 9th Asia-Pacific Software Engineering Conference, 2002.
[5] E. Brynjolfsson, “The Productivity Paradox of Information Technology,” Communications ACM, pp. 66-67, 1993.
[6] Potts, C., "Software-engineering research revisited," Software, IEEE , vol.10, no.5, pp.19,28, Sept. 1993
[7] Scacchi, W., "Managing Software Engineering Projects: A Social Analysis", IEEE Trans. Soft. Engr., SE-10(1), (1984), 49-59.
[8] Scacchi, W., "Understanding Software Productivity", Advances in Software Engineering and Knowledge Engineering, D.Hurley (ed.), Volume 4, pp. 37-70, (1995).
[9] M. Viljan, “Capstone Course on Agile Software Development Using Scrum,” IEEE Transactions on Education, vol. 55, no. 14, p. 99, 2012.
[10] Lakhanpal, B., `Understanding the Factors Influencing the Performance of Software Development Groups: An Exploratory Grou-Level Analysis,' Information and Software Technology, 35(8), (1993), 468-471.
[11] Bendifallah, S. and W. Scacchi, `Work Structures and Shifts: An Empirical Analysis of Software Specification Teamwork', Proc. 11th. Intern. Conf. Soft. Engr., IEEE Computer Society, (1989), 345-357.
[12] G. M. Weinberg, ThePsychology of Computer Programming, 1971.
[13] E. S. Raymond, The Cathedral and the Bazaar, 1999.
[14] Carver, J.; Jaccheri, L.; Morasca, Sandro; Shull, F., "Issues in using students in empirical studies in software engineering education," Ninth International Software Metrics Symposium, 2003. Proceedings. pp.239,249, 3-5 Sept. 2003
[15] L. Adams, “Ten Commandments of Egoless Programming,” Tech Republic, 2002.
[16] Nunnally JC, Psychometric Theory, McGraw-Hill; 3rd edition (January 1, 1994)