by Tony Clear,
Auckland University of Technology

Global Software Engineering and Scaled Agile: Pushing the Boundaries of the Discipline?

Working with my colleague Ramesh Lal recently—researching developments in global software engineering (GSE) supported by scaled agile development methods [10,11]—has raised questions for me about the tunnel vision of our curricula. We need a broader roadmap! In this column, the needs of software professionals working in large scale agile settings are compared to two ACM curriculum models, gaps are identified, and new areas of focus highlighted. This is driven by the arguable premise that most graduates of computer science and software engineering programs are likely to be working in software intensive roles and industries. So how well are we preparing them for those roles?

The reality of modern software engineering practice is that developers increasingly work in global settings and distributed teams [3,4]. Lean and agile practices are becoming more prevalent in both local and global contexts [7,18]. The rise of 'scaled agile' methods [2,11,12,15] is now posing new challenges for practitioners. We are seeing increased use of cross functional teams and "T-skilled" individuals [9] who may have depth in one technical specialty (e.g., Java development, software architecture, or usability engineering), but an accompanying breadth of knowledge to chart roadmaps for products and set these skills in a wider context. Thus, they may become more broadly engaged in collaborative roles, which are customer facing and strategy aware within the wider enterprise setting.

Our current curriculum models [1,16] typically assume single site activities and have a strong focus on the technical aspects of the discipline (whether theoretical or practical). How well do they address these new and evolving needs? Moreover, can they, and should they?

While our graduates need to be technically skilled in these global contexts, they need to move beyond their tunnels with a more rounded set of capabilities. Elements determined to be essential knowledge for software engineers have been identified in the Software Engineering Education Knowledge (SEEK) section of the SE 2014 curriculum model [16]. Yet there is nothing specifically addressing global aspects of professional practice, and only limited coverage of the connections of software development activity to business or product strategy within an organization. With its highly technical focus, the CS 2013 curriculum [1] has far less coverage (e.g., eight curriculum hours allocated to HCI, three hours to software process, two hours to software project management and four hours to requirements engineering). Referring to global software engineering, Matthes [10] in 2011 had observed that universities were struggling with the challenges:

> *When considering the personal requirement today's software engineers are facing in their daily work life, it is surprising to see that teaching GSE at universities is still in its infancy.*

When we ourselves looked specifically at GSE Education courses [4], we saw that:

> *GSE-Ed is found to teach students technical aspects of development but is still falling short in teaching them the many project management, interpersonal and leadership skills required to manage*

the more complex coordination and collaboration processes that are an integral part of GSE.

Yet these are not simply narrow job related capabilities. These missing skills would better equip them for a dynamic future as adaptable and aware global citizens, and enable them to function effectively in a complex professional setting. Driving these needs, there are some critical underlying trends that characterize the incipient combination of *global software engineering* and *scaled agile methods*. First—the need to work globally and cope with the challenges of global distance (*cultural*, *temporal*, and *geographic*) that poses [5,14]; second—the need to work in a more value driven context, where the purpose of the enterprise, the goals of the customers, and the focus of the software need to more tightly align [6,10]; and third—the concurrent need for developers to be more proficient with working in multi-skilled teams, collaborating with customers and colleagues across the enterprise [11].

The scaling of agile methods now opens an opportunity for software developers to become a more integral part of their enterprises, especially in vendor organizations, contributing as innovation partners to the start of the requirements pipeline, through product strategy, vision, roadmaps, prioritization, conceiving feasible design solutions, thus serving customer needs and addressing their customer experience through implementing well targeted, high quality, working software through regular releases. This moves computing well beyond the notion of mere *problem solving* where "*students apply their knowledge to specific, constrained problems and produce solutions*." [17] For in that myopic model of the discipline or profession, the problem-posing and problem-framing does not include the input of skilled software engineers, who of necessity must relegate those responsibilities to others. In turn, higher level managers dismissively ignoring the input of technical experts in the developing of software intensive products and services has caused waste and failure in many projects [11].

The evolving nature of professional practice in GSE, now demands explicit awareness of the following new dimensions: 1) the tight interrelationship between software process, practices, and roles brought about by lean and agile approaches; 2) strategies and techniques for effective global collaboration; and 3) an appreciation of the interactions between software and business functions, processes and personnel

demanded by enterprise forms of agility. So, I argue here that these higher-level portfolio and program level topics, which move the notion of agility beyond the team level, in combination with approaches to multi-skilled and global team collaboration, need to be incorporated into computing curricula. This charts a new roadmap for computing curriculum models!

We need more broadly educated, yet technically able, graduates. Those who can see the broader purpose and take global leadership roles in partnership with other stakeholders in their enterprises will have more to offer and have a stronger chance of survival in tomorrow's shifting global settings. Such graduates will not only be more effective and contributing citizens, but better equipped to deal with the realities of their professions. We owe it to them to both deepen and broaden our curriculum to better face these challenges. ❖

References

1. ACM/IEEE CS Joint Task Force on Computing Curricula. *Computer Science Curricula 2013*. (ACM Press and IEEE CS Press, 2013); doi: http://dx.doi.org/10.1145/2534860.
2. Ambler, S.W. and Lines, M. Disciplined agile delivery: *A practitioner's guide to agile software delivery in the enterprise*. (IBM Press, New York, 2012).
3. Aspray, W., Mayadas F. and Vardi M. *Globalization and Offshoring of Software - A Report of the ACM Job Migration Task Force*. (ACM, New York, 2006).
4. Beecham, S., Clear, T. and Noll, J. Do We Teach the Right Thing? A Comparison of Global Software Engineering Education and Practice, in *Proceedings 2017 IEEE 12th International Conference on Global Software Engineering*, edited by D. Cruzes and A. Sharma (IEEE, Los Alamitos, California, 2017), 11–20.
5. Clear, T., Beecham, S., Barr, J., Daniels, M., McDermott, R., Oudshoorn, M., Savickaite, A., and Noll, J. Challenges and Recommendations for the Design and Conduct of Global Software Engineering Courses: A Systematic Review, in *Proceedings of the Working Group Reports of the 2015 on Innovation & Technology in Computer Science Education Conference*, edited by N. Ragonis, and P. Kinnunen (ACM, New York, 2015), 1–39.
6. Dingsøyr, T. and Lassenius, C. Emerging themes in agile software development: Introduction to the special section on continuous value delivery. *Information and Software Technology*, 77 (2016), 56–60.
7. Hanssen, G., Šmite, D. and Moe, N. Signs of Agile Trends in Global Software Engineering Research: A Tertiary Study. in *Sixth IEEE International Conference on Global Software Engineering Workshops*. (IEEE, Los Alamitos, California, 2011).
8. Kirk, D. and Tempero, E. A lightweight framework for describing software practices. *Journal of Systems & Software*, 85, (2012), 582–595.
9. Kloppenborg, T., Forte, F. and Lensges, M. Identifying key Agile behaviors that enhance traditional project management methodology, in *2017 Annual Meeting of the Decision Sciences Institute Proceedings*, edited by X. Zhao (Decision Sciences Institute, Washington, D.C., 2017), 12916411–129164122.
10. Lal, R. and Clear, T. Enhancing product and service capability through scaling agility in a global software vendor environment [forthcoming], in *Proceedings 2018 IEEE 13th International Conference on Global Software Engineering*, edited by R. Espinoza and D. Šmite, (IEEE, Los Alamitos, California, 2018).
11. Lal, R. and Clear T. Scaling Agile at the Program Level in an *Australian Software Vendor Environment: A Case Study, in Australasian Conference on Information Systems*. (AIS, 2017), 1–12; https://www.acis2017.org/wp-content/uploads/2017/11/ACIS2017_paper_184_FULL.pdf. Accessed 2018 February 12.
12. Leffingwell, D. *SAFe® 4.0 Reference Guide: Scaled Agile Framework® for Lean Software and Systems Engineering*. (Addison-Wesley Professional, Indianapolis, 2016).
13. Matthes, F. et al. Teaching Global Software Engineering and International Project Management-Experiences and Lessons Learned from Four Academic Projects, in *Proceedings of the 3rd International Conference on Computer Supported Education* (CSEDU, 2011).
14. Noll, J., Beecham, S. and Richardson, I. Global Software Development and Collaboration: Barriers and Solutions. *ACM Inroads*, 1, 3 (2010), 66–78.
15. Paasivaara, M. Adopting SAFe to scale agile in a globally distributed organization. In *Proceedings of the 12th International Conference on Global Software Engineering*, edited by A. Sarma and D. Cruzes, (IEEE Press, Los Alamitos, California, 2017), 36–40.
16. The Joint Task Force on Computing Curricula, *Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*, (ACM, New York, 2015), 134.
17. Tucker, AB. Computing Curricula 1991, *Communications of the ACM*, 34, 6 (1991), 68–84; doi:10.1145/103701.103710.
18. Wang, X., Conboy, K. and Cawley, O. "Leagile" software development: An experience report analysis of the application of lean approaches in agile software development. *Journal of Systems and Software*, 6, (2012), 1287–1299.

Tony Clear
School of Computing and Mathematical Sciences
Auckland University of Technology
Private Bag 92006
Auckland, 1142 New Zealand
Tony.Clear@aut.ac.nz

DOI: 10.1145/3233988