Global Software Development for the Enterprise

Rajiv Ramnath
CETI
Department of Computer Science and Engineering
The Ohio State University
Ramnath.6@osu.edu

Abstract

In this position paper, we present certain observed characteristics of global software development for the enterprise, as well as trends in enterprise information technology needs for a global enterprise. We then identify IT workforce needs and the consequent curriculum elements desired.

1. Introduction

From 1998 to 2003, I led the product development group for a workflow company - Concentus Technology Corp. - located in Columbus, Ohio. Our global team grew to 8 in Columbus, one each in Toronto, Canada and Chicago, Ill. and 20 within an offshore outsourcing partner in Bangalore, India, with the goal of "24-hr joint product development". Since moving to academia in 2003, I teach the senior software engineering Capstone course where we design, engineer and implement pieces of enterprise scale systems, and consumer-directed application prototypes. Also, through CETI, teams of students and faculty conduct collaborative research projects sponsored by local industry (primarily large regional, national and multinational firms, "consumers" of information technology), in technology strategy, technology management, and enterprise and system architecture. We also conduct a CIO-level forum on Enterprise Architecture where the state-of-the-practice on topics such as standards adoption, governance, the role of the architect, and education needs of professionals and entry-level students are presented and discussed, using internal examples provided by the companies. This position paper is based on these experiences.

2. The Nature of Global Software Development for the Enterprise

Here I list social, organizational, and technical observations from my global software development experiences:

- Remote team members developed complete and rich electronic personas. That these were often different from their actual, physical personas was not the issue or a limitation; instead the existence of these personas meant that real chemistry was created.
- Once organizational structures stabilized and technical-level interactions became dominant, trust became a matter of fact. Trust remained as something that could be lost at an individual-to-individual level, but this loss of trust was not correlated to distance.
- The remote team was organically grown from a proof-of-concept trial with a small team of 4 to a final scale of approximately 30. Our experiences were strongly impacted by scale - in other words, there were different experiences at each level of scale – from a small development team to a large team to multiple teams with compartmentalized responsibilities. However, early emergence of a stable leadership group (composed of members on both sides) helped resolve most issues.
- Distance did bring with it lack of visibility into individual performance.
- Identity and loyalty to the team (as opposed to loyalty towards the separate organizations) was created and critical to success. This identity was constantly under stress, especially with leadership turnover.
- Cultural, social and communication issues were quickly overcome and rarely at issue once a common technical and domain vocabulary was established.
- Understanding domain requirements as well as the complete architecture of the product lines being developed needed to be a constant focus; but this was true both within the on-shore product group as well as the offshore one. However, unfiltered and direct access to the voice of the customer was a factor of location, and therefore an issue.
- Early establishment of architecture, design and process standards was critical.
- Early end-to-end validation of work-products was critical, and counter to the conventional wisdom for outsourcing of creating complete specifications to be thrown over the wall to be implemented. However, establishment of complete work-product *standards* was essential for clear communication.
- Access to a representative development infrastructure at all locations was an issue.

Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06)
0-7695-2655-1/06 $20.00 © 2006 IEEE

3. Trends in Enterprise IT Requirements and Expectations

Within enterprise IT, several requirements and issues emerge, as follows:

- There exists a tension between IT and the dynamics of the business – such as a demand for rapid, end-user-driven change at the edges, and compatibility with a unified enterprise and technology architecture.
- The need of the hour is technology-enabled business innovation - e.g. by a Service Oriented Architecture.
- All technology projects must have a clearly defined *business* ROI.
- Globalization of the business implies globalization of the enterprise software – its interface, information and embedded processes and its development.
- Technology departments need extensive enterprise knowledge in order to be effective – from the target market of the business to organizational structure and business processes and legacy environment.
- The need is for planned variation and adaptation to the environment – legacy infrastructure as well as business circumstance. For example, a single "best-practice" SDLC methodology does not exist, but must be tuned to project "shape" [4].
- A plethora of enterprise technology exists, and is applied, as well as universally available open-source technology.

4. IT Workforce Needs of the Future

The IT workforce must understand the characteristics of remote software development and the *evolving* IT needs of the enterprise, and learn to be responsive to trends in these areas. Specifically, the workforce must understand the *organizational* dynamics of trust building, identity and loyalty, and issues related to evolving and changing organizational scale. Next must come an understanding of the goals of the *business* and *domain* to which the software development is targeted. Finally, must come *technical* and software development lifecycle (SDLC) expertise – an understanding of architecture, of good design and implementation practices, the appropriate application of software engineering methodologies based on project shape, and technology and infrastructure management strategies, all to meet business goals.

5. Curriculum Development Challenges

As detailed in [1] the education of computing has evolved into multiple intersecting disciplines (computer engineering, computer science, software engineering, information systems, and information technology) but with different focuses. The predominant computing discipline offered in the US is the Computer Science (CS) degree that is shaded towards the theory and fundamentals of computing in its emphasis. Enterprise software development however requires a holistic approach that spans these disciplines and further adds a business dimension as well as a domain discipline dimension. Global software development further adds cultural, societal and regulatory dimensions to the mix of skills needed.

Providing a complete education in all the computing disciplines, cultural and social dimensions and all possible domains is obviously impossible. Thus, the focus should be on inculcating:

- *Systems-level* thinking and understanding, on the static, and especially the *dynamic* aspects of systems.
- A *process* of lifelong learning

From a computing point of view, one key to the above is developing an ability to abstract and *capture* models of systems, processes, and organizations, even goals, motivations, and strategies, and their *variations*, in particular through semantic representational techniques – such as ontologies, domain and feature modelling, patterns, product line techniques and so on, using notations such as UML and its extensions, Where appropriate, representational techniques must be borrowed from other disciplines.

At this point, standard classroom curriculum has not been developed to cover this range of skills, nor will it ever be. Therefore curriculum has to have a significant *experience-based* component where knowledge frameworks can be applied, adjusted and learned from in a reflective [2] and empirical [3] manner within a microcosm of the real world. What should be emphasized is not the experiential learning itself but the *process* of discovery, application and adjustment and the development of a *vocabulary* of communication.

Finally, this education should be in addition to, and not in place of, the development of fundamental communication, mathematical and computing skills.

7. References

[1] http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf

[2] Schon, Donald, "The reflective practitioner: How professionals think in action," Basic Books, ISBN 0465-06878-2, 1983

[3] Wegner, P., Goldin, D., "Principles of Problem Solving," Viewpoint, CACM, July 2006.

[4] Boehm, B., Turner, R. "Balancing Agility and Discipline – A Guide to the Perplexed", Pearson Education, ISBN 321-18612-5, 2004.