UACEE International Journal of Advances in Computer Science and its Applications – IJCSIA
Volume 3 : Issue 2 [ISSN 2250 – 3765]
Publication Date : 05 June 2013

Global Prospect of Distributed Agile Software Development: A Review

Anurag Tiwari, Praveen Kumar Patel, Manuj Darbari

***Abstract* — The entangled and rival market situation has resulted the distributed software development maximise their reach with new emerging technologies that most of companies vastly applying with their processes is global software development. Companies need to utilise their existing resource as much as possible. Thus the ability to collaborate thoroughly has become a critical factor. Considerable geographical distance, dissimilar time zone, mismatch process between the team and cultural adverseness in distributed software development leads to collaboration & communication which contrariwise effect the project. Using the agile practices provide fast, light and efficient than any other vigorous method without being chaotic of distributed development .This paper examine the intersection of these two momentous trends for software product development i.e. agile practices and distributed software development .we discussed DSD challenges based on their root cause, which helps structuring a efficacious distributed development team .**

Keywords- Agile practices, Distributed software development, Global software development, distributed agile software development.

I. *INTRODUCTION*

There is a growing trend towards working with distributed teams across the globe. Many factors are responsible for this trend. Some of the prominent ones are increasing cost of maintaining physical offices with a parallel pressing need to cut down costs, success of outsourcing as a business model, availability of internet-enabled smart devices, deeper penetration of broadband and of course the advent of cloud computing.

Anurag Tiwari , M.Tech Student
Department of Computer Science
Babu Banarasi Das University
Lucknow,India
anuragrktiwari@gmail.com

Praveen Kumar Patel, M.Tech Student
Department of Computer Science
Babu Banarasi Das University,
Lucknow,India
praveenpatelsiet@gmail.com

Dr. Manuj Darbari
Associate. Prof Department of Computer Science
Babu Banarasi Das University,
Lucknow,India
manuj_darbari@acm.org

The combination of above factors (in any permutation) clearly indicates that future will see more organizations romancing the distributed teams to a large extent.In the new era of software development every software development companies has pressure to provide better product in lesser timeframe, so every development houses need to utilise their most of the resource at one time span. So the problem that most of development housed faces is how to exploit most of their onsite and offsite resources efficiently.Thus the most of the software vender utilise mechanism of distributed software development (DSD) to solve out their development problem. It enhances the better software development in rapid way and access to skilled of resource. Their main objective is to develop high quality products at lower cost than co-located developments by optimizing the resources [1]. They need to employ resource on global prospective from different location within the company itself and partner companies all over the world. In fact, Global software development of software intensive systems is the reality in most software projects: up to 80 or 90 percent of software projects are now globally distributed and distributed development (e.g., outsourcing, one mode of distributed development) will continue to grow (Fryer & Gothe, 2008, Forrester, 2010). So the current market situation and economic condition has meant that efficiency of product development and control over the product life cycle cost is even more important for companies.

Software is developed in a multi-site, multicultural, globally distributed environment. Engineers, managers and executives face formidable changes on many levels, from technical to social and cultural [2].Globally distributed software engineering also allows organisations to benefit from access to a larger qualified resource pool with the promise of reduced development costs. Another potentially positive impact of globally distributed engineering is innovation, as the mix of developers with different cultural backgrounds may trigger new ideas (Ebert & De Neve, 2001; Herbsleb & Moitra, 2001; Damian et al., 2004). According to the industrial practice inventory [3], the most common reasons for collaboration were to reduce development costs, to acquire competence (technology competence or knowledge of a specific market) and to avoid investing in a company's non-core competence areas. Further reasons included potential timesaving, the establishment of new business opportunities with new partners, flexibility with respect to the number of in-house resources and overcoming problems of availability of in-house resources. Likewise, in last decade agile software development is the very good choice for software development community .Agile software development methodologies have been greeted with enthusiasm by many software developers because work is done at different levels in parallel [4]. We can use our creativity on the design level you can also have the fun of implementing the design in a smart way. Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams [5]. In *The 2008 State of Agile Development* survey, conducted by Version One, 57% of

respondents stated that their teams were distributed. Furthermore 41% of respondents said that they were currently using, or plan to combine, agile with outsourced development. When faced with these sorts of numbers it seems that the agile practice of placing the entire team in a single room is at odds with what's actually going on within a large part of the software development community. While collocating your team is clearly the recommended approach many teams are unable to do this and are faced with trying to stick to their agile principles and apply agile practices to distributed development.

"High-bandwidth communication is one of the core practices of Scrum… The best communication is face to face, with communications occurring through facial expression, body language, intonation, and words. When a white board is thrown in and the teams work out design as a group, the communication bandwidth absolutely sizzles." Ken Schwaber, the Enterprise and Scrum [6].

Communication represents a significant part of the effort involved in delivering software, so lowering the barriers to communication increases a team's overall efficiency [7]. Collocation also reduces project risk [8]. In a recent survey of agile projects, success rates for teams located in the same room were over 20% higher than those for geographically distributed teams [9]. Its main impact on the software development life cycle, delivered to the costumer. In general software engineering involves the design, development, maintenance and documentation of software systems. Agile methods works very well in highly dynamic business and IT environment. Many organizations that develop software using Agile have started looking for skills and talent available at much lower rates and are anxious to source the development work to these centers [10].

This paper focuses on understanding the accomplishment s achieved by adopting agile practices with distribute software development (DSD).what are the emerging challenges which agile distributed teams have to face. if any, then find out the way to overcome such type of problem.

This paper is organised as follows: section II present the current scenario of distributed software development/Global software development .Section III explain the agile methodologies. This is followed by Section IV which discusses in details the benefits and challenges faced by distributed agile teams. In section V the problem and list out the various techniques that can help to overcome the challenges put by agile technique which combine with distributed development and in final section conclude the paper.

II. Global *software development*

IBM and a few other major computer vendors have had globally distributed software development since the 1960's [12]. Moreover, the current dynamic business environment requires organizations to develop and evolve software systems at Internet speed. As a consequence of these major trends, software development organizations have been striving to blend agile software development methods like Extreme Programming and distributed development to reap the benefits of both. However, agile and distributed development approaches differ significantly in their key tenets [11].Some large multinational corporations have been developing software in a distributed manner even before the term GSD was formulated [12].Another form of distributed software development called contract programming emerged in 1970's [13]. Contract programming was an example of partial outsourcing where certain parts of development work was transferred to a third-party service provider [13]. The globalization of software began in the 1990's with the PC revolution, which changed the culture of software development [2],[7]. Software development is no longer the domain of just large companies, but also smaller sized companies [2].

The irreversible wave of globalization has amplified the competition in software industry. However with limited access skilled labor and grueling time to market many software companies started to look for partners and set up development sites in different countries [2],[14]. This revolution resulted in multi-site, multi-cultural, and globally distributed software development [15]. The concept of GSD is derived from contract programming which a form was of outsourcing in the 1970's [13]. However contract programming was limited only to application programming. GSD is not limited to outsourcing and programming tasks only.GSD is defined by Sangwan et al [1] as "software development that uses teams from multiple geographic locations"[16].

The collaboration modes among these development teams can be either in outsourcing or offshoring mode. Outsourcing (inter-organizations) refers to subcontracting a process (i.e. developing a product) to an external organization or third-party. Offshoring (intraorganization) refers to the relocation of a business process by a company to another country [16].

GSD is different from "normal" or collocated development [17]. Apart from the different collaboration modes, GSD has characteristics that are different compared to collocated development. GSD has attributes which are called global factors. Global factors are the environmental factors which are unique to GSD, and these are [18], [19]:

1. **Multisourcing –** multiple distributed member involvement in a joint project, characterized by a number of collaboration partners.
2. **Geographic distribution –** partners are located far away from each other.
3. **Temporal diversity** – characterized by the level of working hours overlay.
4. **Socio-cultural diversity** – level of social, ethnic, and cultural fit.
5. **Linguistic diversity** – characterized by the level of language skills.
6. **Contextual diversity** – level of organizational fit (diversity in process maturity and work practices).

III. Agile Software Development

Before going to discuss about the distributed agile software development we have to know some brief overview of premier agile methodologies. Agile software development methods emerged in the late 1990s [20]. The term 'Agile' was adopted as the umbrella term for methods such as Scrum [21], XP (eXtreme Programming) [22], Crystal [23], FDD (Feature-Driven Development) [24], DSDM (Dynamic Software Development Method) [25], and Adaptive Software Development [26]. Scrum and XP are considered to be the most widely adopted Agile methods in the world [27]. XP focuses on developmental practices, while Scrum mainly covers project management [28]. Agile teams are self-organizing teams [29,30,31,32–33] composed of ''individuals [that] manage their own workload, shift work among themselves based on need and best fit, and participate in team decision making.'' [34]. Takeuchi and Nonaka [35] describe self-organizing teams as exhibiting autonomy, cross-fertilization, and self transcendence.

Self-organizing teams must have common focus, mutual trust, respect, and the ability to organize repeatedly to meet new challenges [20]. Self-organizing teams are not leaderless, uncontrolled teams [26, 35, 36]. Leadership in self-organizing teams is meant to be light-touch and adaptive [37], providing feedback and subtle direction [38, 39, 35]. Leaders of Agile teams are responsible for setting direction, aligning people, obtaining resources, and motivating the teams [38]. Agile projects have job titles such as Scrum Masters [21] and (XP) Coaches [41] instead of traditional managers.Agile methods advocate high levels of collaboration between the custormer in order to frequently release product features that deliver business value in each iteration. Several Agile practices demand customer collaboration such as planning, prioritizing, reviewing, and providing feedback. In Scrum, the customer is an important part of the process [21].

The customer representative, also known as Product Owner, is responsible for defining the features of the product; prioritizing the list of features (backlog); reviewing the developed features; and providing feedback to the team. The development team is meant to use the backlog to develop corresponding features every iteration in close collaboration with the customer [21]. Similarly, XP, has a established customer role [22] that is responsible for providing and prioritizing requirements; assessing the developed tasks against a set of customer-defined acceptance criteria; and providing frequent feedback to the team [22]. Agile methods can be beneficial when combined with distributed development. There are studies which show that agile principles help in overcoming some challenges faced by distributed development [41], [42], [43], [44], [45]. Agile when combined with DSD also brings some new challenges which will be discussed further.

IV. Benefits obtained by combining agile with Distributed Development

Distributed software development seems to cause decreased visibility of project stutus and agile process based on short continuous iterations make it easier to see the problems already on early stages of the project, Continous intergration of software code,which is a central part of agile methods,also helps to reduce configuration management issues. Use of agile principles seems to have a positive effect on communication between teams as development in cycles makes it easier for participants to see the short term goals [46].Sprint reviews can be an effective way to improve external communication while they help to share information about the feature and requirement dependencies between stakeholders.

Agile development processes place both added importance as well as special demands on your software quality assurance (SQA) practices. As an integrated part of the agile team, testers participate in the full life-cycle from requirements through release [47]. While many of the principles and practices of software quality assurance apply, an agile approach requires some new ways of viewing testing activities in the development process. Successful outsourcing projects are the ones that strike a good balance between testing and quality assurance throughout the lifecycle of the project [48].

Agile principles can even help create trust between different cultures involved in the process by constant communication and delivery of software [49]. According to a study made by Passivara, Durasiewicz and Lassenius quality of software and communication are improved and communication and collaboration is more frequent than before because of the Scrum methodology used in the project. Also the motivation of team members was reported to have increased [50]. Thus, agile in distributed teams has proved to be beneficial for project's quality and performance.

V. *Best open source tools suited for distributed agile development*

Many tools exist that can be used to support distributed development. These are collaborative modeling tools, collaborative writing tools, conferencing tools, virtual meeting tools, and so on. Most of these tools used in XP software development are either free or inexpensive. Automated builds provide teams with comprehensive up-to-date status of their project, what tasks have been completed, what tasks remain, whether the current work meets test requirements, and whether all the moving parts are working together properly. Any member of the team can get this information at any time, greatly simplifying communications and ensuring that everyone understands the state of the project.

[A] Tools for Synchronous and Asynchronous Communication

Synchronous communication (SC) can be defined as real-time communication that occurs between two or more people through a virtual or electronically mediated system. SC systems have become more prevalent as high-speed network connections has increased and become both more readily available and affordable. Synchronous communication tools are interactive programs that allow two or more users to communicate by means of textual, audio, video messages or graphics messages, such as virtual whiteboard for brainstorming. Instant Messaging tools (IMs) show team member who is online and allow a real time communication.

Asynchronous communication tools allow more users to exchange information in not interactive ways that is they don't keep busy who communicates in order to wait a response. An example of asynchronous communication is the use of a blackboard where more users can hang" messages. These tools suffer from real communication but they don't require all users take part in the project simultaneously. Moreover, typically all messages are stored and available to the next consultation.

V. CONCLUSION AND FUTHURE SCOPE

The After examine the many circumstances along with the benefit there are many challenges that comes when we bind agile with distributed software development. Although there are business reasons for distributing a team, distribution leads to team dysfunction by inhibiting communication. Agile teams rely on intense person to person communication, both with team and the customer.Thus, these two trends i.e. distributed development and agile approach face difficulty when it comes to compatibility issues. Indentify and manage key resources is mean the idea is not that distributed teams should follow a particular organizational or governance structure. The important point is that all team members understand point is that all team is organized and now their role is expected to interact with others maintaining consistency in team structures and management structures avoids confusion.Each phase of the software development process comprises different communication flows and challenges. In distributed software development normal communication lags or delays are compounded by time zone differences. Informal communication which works well in collocated agile teams is not possible in distributed teams. This also leads to lack of trust amongst the team members. Thus, extra efforts are required on behalf of the team members to maintain effective communications. Scheduling meetings become more difficult due to limited overlapping work hours.

There is a need to develop methodologies to add intelligence to information via accepted metadata and other standards. Collected for one purpose so that it can also be used as proxy in another policy area, and develop dynamic monitoring systems. Increase in the amount of documentation also helps in achieving better co-ordination requirement clarification with use-cases and user-stories [51]. Thus, flexibility and adjustment of project is recommended instead of strictly following agile principles [52].Efforts should be made bring the distributed team members together at the start, end and at other pivotal points during the project. This helps in building shared understanding of the problem domain and also working relationships within the team [53].

So, in future need to established high end dedicated communication channel which help provide better management and developer communication across the employ resource on global prospective from different location within the company itself and partner companies all over the world. and by adopting best management practices that can improve communication and process overhead.

REFERENCES

[1] Migues Jimeenez, Mario Piattini , Aurora Vizcanio, “Review article Challenges and Improvements in Distributed Software Development: A Systematic Review”, Hindawi Publishing Corporation, Advances in Software Engineering, Volume 2009, Article ID 710971.

[2] J.D Herbsleb and D. Moitra (2001), ”Global Software Development, IEEE Software, March/April, USA, p. 16-20, http://conway.isri.cmu.edu/~jdh/collaboratory/research_papers/IEEE_SW_editorial_final.pdf (Retrieved on March 12, 2010 from) .

[3] Päivi Parviainen ,“Collaborative embedded systems development Survey of state of the practice”In: Proceedings of the 13th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS), March 27–30,2006, Potsdam, Germany.

[4] Robert C. Martin, Agile Software Development, Principles, Patterns, and Practices, Prentice Hall, 2002

[5] “Agile software development”, http://en.wikipidea.org/wiki/agilesoftware Development. Retrieved on Feb.2013.

[6] Ken Schwaber, The Enterprise and Scrum, Microsoft Press, ISBN 073561993X

[7]Teasley, Covi, Krishnan &Olson, How Does Radical Collocation Help a Team Succeed? http://possibility.com/Misc/p339-teasley.pdf

[8] David Norton, Making the Most of 'Agile' in a Distributed Development Environment, Gartner Research G00159246, August 2008

[9] Scott Ambler, Has Agile Peaked?, Dr. Dobbs Journal, June 2008.

[10] M.Simon, ”Internationally Agile”, http://www.informit.com/articles/article.aspx?p=25929, March 15, 2002 (Retrieved on March 10, 2010)

[11] balasubramaniam ramesh, lan cao, kannan mohan, And peng xu, "can distributed software Development be agile?"In: **communications of the ACM** October 2006/Vol. 49, No. 10.
[12] R. Prikladnicki, J. Nicolas Audy, and R. Evaristo, "A Reference Model for Global Software Development: Findings from a Case Study," *2006 IEEE International Conference on Global Software Engineering (ICGSE'06)*, 2006,pp. 18-28.
[13] D. Damian, L. Izquierdo, J. Singer, and I. Kwan, "Awareness in the wild: Why communication breakdowns occur," *International Conference on Global Software Engineering, ICGSE 2007, August 27, 2007 - August 30, 2007*, Munich, Germany: Inst. of Elec. and Elec. Eng. Computer Society, 2007, pp.81-90.
[14] J. Herbsleb and A. Mockus, "An empirical study of speed and communication in globally distributed software development," *Software Engineering, IEEE Transactions on*, vol. 29, 2003, pp. 481-494.
[15] P. Vlaar, P. van Fenema, and V. Tiwari, "Cocreating understanding and value in distributed work: How members of onsite and offshore vendor teams give,make, demand, and break sense," *MIS QUARTERLY*, vol. 32, Jun. 2008, pp.227-255.
[16] R. Sangwan, M. Bass, N. Mullick, D.J. Paulish, and J. Kazmeier, *Global Software Development Handbook (Auerbach Series on Applied SoftwareEngineering Series)*, Auerbach Publications, 2006.
[17] E. Carmel, *Global software teams: collaborating across borders and time zones*, Prentice Hall PTR, 1999.
[18] D. Šmite and J. Borzovs, "A framework for overcoming supplier related threats in global projects," *Software Process Improvement*, 2006, pp. 50–61.
[19] N.B. Moe and D. Smite, "Understanding a lack of trust in global software teams: A multiple-case study," *Software Process Improvement and Practice*,vol. 13, 2008, pp. 217-231.
[20] C. Larman, V.R. Basili, Iterative and incremental development: a brief history,Computer 36 (2003) 47–56.
[21] K. Schwaber, M. Beedle, Agile Software Development with SCRUM, Prentice Hall, 2002.
[22] K. Beck, Extreme Programming Explained: Embrace Change, first ed., Addison Wesley Professional, 1999.
[23] A. Cockburn, Crystal Clear: A Human-Powered Methodology for Small Teams, Addison-Wesley Professional, 2004.
[24] S. Palmer, M. Felsing, A Practical Guide to Feature-Driven Development, Pearson Education, 2001.
[25] J. Stapleton, Dynamic Systems Development Method, Addison Wesley, 1997.
[26] J. Highsmith, Adaptive Software Development: A Collaborative Approach to Managing Complex Systems, Dorset House Publishing, New York, 2000.
[27] S. Fraser, R. Reinitz, J. Eckstein, J. Kerievsky, R. Mee, M. Poppendieck, Xtreme programming and agile coaching, in: OOPSLA Comp.03, ACM, New York, 2003, pp. 265–267.
[28] T. Dybå, T. Dingsoyr, Empirical studies of agile software development: a systematic review, Inf. Softw. Technol. 50 (2008) 833–859.
[29] T. Chow, D. Cao, A survey study of critical success factors in agile software projects, J. Syst. Softw. 81 (2008) 961–971.
[30] A. Cockburn, J. Highsmith, Agile software development: the people factor, Computer 34 (2001) 131–133.
[31] J. Highsmith, M. Fowler, The agile manifesto, Softw. Dev. Mag. 9 (2001) 29–30.
[32] R. Martin, Agile Software Development: Principles, Patterns, and Practices, Pearson Education, NJ, 2002.
[33] H. Sharp, H. Robinson, Collaboration and co-ordination in mature extreme programming teams, Int. J. Hum.–Comput. Stud. 66 (2008) 506–518.
[34] J. Highsmith, Agile Project Management: Creating Innovative Products, Addison-Weasley, USA, 2004.
[35] H. Takeuchi, I. Nonaka, The new product development game, Harvard Bus. Rev. 64 (1986) 137–146.
[36] A. Cockburn, J. Highsmith, Agile software development: the people factor, Computer 34 (2001) 131–133. [37] S. Augustine, B. Payne, F.
[38] L. Anderson, G. Alleman, K. Beck, J. Blotner, W. Cunningham, M. Poppendieck, R. Wirfs-Brock, Agile management – an oxymoron: who needs managers anyway?, in: OOPSLA '03, ACM, New York, 2003, pp 275–277.
[39] T. Chau, F. Maurer, Knowledge sharing in agile software teams, 2004.
[40] S. Fraser, R. Reinitz, J. Eckstein, J. Kerievsky, R. Mee, M. Poppendieck, Xtreme programming and agile coaching, in: OOPSLA Comp.03, ACM, New York, 2003, pp. 265–267.

[37] S. Augustine, B. Payne, F. Sencindiver, S. Woodcock, Agile project management: steering from the edges, Commun. ACM 48 (2005) 85–89.
[38] L. Anderson, G. Alleman, K. Beck, J. Blotner, W. Cunningham, M. Poppendieck, R. Wirfs-Brock, Agile management – an oxymoron: who needs managers anyway?, in: OOPSLA '03, ACM, New York, 2003, pp 275–277.
[39] T. Chau, F. Maurer, Knowledge sharing in agile software teams, 2004.
[40] S. Fraser, R. Reinitz, J. Eckstein, J. Kerievsky, R. Mee, M. Poppendieck, Xtreme programming and agile coaching, in: OOPSLA Comp.03, ACM, New York, 2003, pp. 265–267.
[41] M. Fowler, " Using an Agile Software Process with Offshore development", http://martinfowler.com/articles/agileOffshore.html, July 2006 (Retrieved on March 10, 2010).
[42] H.Smits, "Implementing Scrum in a Distributed Software Development Organization", Agile 2007, p.371-375, IEEE.
[43] M.Kircher, P.Jain, A.Corsaro, D.Levin, "Distributed eXtreme Programming", Proceedings of the International Conference on eXtreme Programming and Agile Methods, p.147-154, 2004
[44] K.Sureshchandra, J.Shrinivasavadhani, " Adopting Agile in Distributed Development", IEEE International Conference on Global Software Engineering , p.217-221, 2008.
[45] J.Sutherland, A.Viktorov, J.Blount, N.Puntikov, *Distributed Scrum: Agile Project Management with Outsourced Development Teams*, International Conference on System Sciences, p.274a-274a, IEEE, 2007.
[46]M.Pikkarainen, J.Haikara, O.Salo, P.Abrahamson, J.Still, "The impact of agile practices on communication in software development", Empir. Software Eng, 13:303-337, 2008.
[47] Robert C. Martin, Agile Software
Development, Principles, Patterns, and Practices, Prentice Hall, 2002[48] Anderson, David J., and Eli Schragenheim, Agile Management for Software Engineering: Applying the Theory of Constraints for Business Results, Prentice Hall, 2003.
[49] M.Paasivaara, C.Lassenius, "Could Global Software Development Benefit from Agile Methods?" International Conference on Global Software Engineering 2006, p.109- 113, IEEE
[50] M.Paasivaara, S. Durasiewicz, C.Lassenius, Using Scrum in Distributed Agile Development: A Multiple Case Study, IEEE International Conference on Global Software Engineering , p.195-204, 2009.
[51] M.Cataldo, M.Bass, J.D.Herbsleb, L.Bass, "On Coordination Mechanisms in Global Software Development", IEEE Conference on Global Software Engineering , p.71-80, 2007
[52] H. Holmstrom, B. Fitzgerald, P. J. Agerfalk, E. O. Conchuir, "Agile Practices Reduce Distance in Global Software Development" Information Systems Management, Summer, pp. 7- 26, 2006
[53] Ade Miller," Distributed Agile Development at Microsoft patterns and practices", Microsoft patterns and practices, http://www.pnpguidance.net/Post/DistributedAgileDevelopmentMicrosoftPatternsPractices, October 2008. (retrieved on March 9, 2010)