Literature Rview: Distributed Agile Teams, problems and solutions

Stevenson Gossage

Carleton University, Shcool of Computer Science, Ottawa, Canada
{sgossage,lnbip}@connect.carleton.ca
http://www.scs.carleton.ca/

Abstract. Literature review stuff here.

Key words: Agile, Planning, Design, Multitouch, Distributed,collaboration, team

1 Beyond models and metaphors: visual formalisms in user interface design

1.1 Author

Nardi, B.A. and Zarmer, C.L.

1.2 abstract

The user interface has both syntactic functions-supplying commands and arguments to programs-and semantic functions-visually presenting application semantics and supporting problem solving cognition. The authors argue that though both functions are important, it is time to devote more resources to the problems of the semantic interface. Complex problem solving activities, e.g. for design and analysis tasks, benefit from clear visualizations of application semantics in the user interface. Designing the semantic interface requires computational building blocks capable of representing and visually presenting application semantics in a clear, precise way. The authors argue that neither mental models not metaphors provide a basis for designing and implementing such building blocks, but that visual formalisms do. They compare the benefits of mental models, metaphors and visual formalisms as the basis for designing the user interface, with particular attention to the practical solutions each provides to application developers. [?]

2 Application of cognitive theories and knowledge management to requirements engineering

2.1 Author

White, S.M.

2.2 abstract

In this paper, we discuss the systems engineering process during front-end phases, with emphasis on capturing requirements related knowledge by inter-organizational and systems of systems teams. We discuss four dimensions of knowledge identified in the knowledge management literature, and apply them to the capture of requirements related knowledge. We discuss and compare three cognitive theories (Situated Action, Activity Theory, and Distributed Cognition) and examine them with respect to improving system requirements engineering and analysis. We find that research in KM and cognition offers significant ideas and insights for improving the requirements process. [19]

3 Challenges and Solutions in Distributed Software Development Project Management: A Systematic Literature Review

3.1 Author

da Silva, F.Q.B. and Costa, C. and Frana, A.C.C. and Prikladinicki, R.

3.2 abstract

This paper presents a systematic literature review of the challenges, best practices, models, and tools in Distributed Software Development (DSD) Project Management. The objective is to collect and systematize reported knowledge in terms of what are the difficulties in managing DSD projects, what are the best practices to overcome these difficulties, and how existing models and tools support these practices. We found 54 works related to DSD project management, published between 1998 and 2009. Using the data systematically extracted from these works, we propose an evidence-based DSD project management improvement model. Our contention is that this model can support practitioners and researchers to better understand the landscape of DSD project challenges and devise more effective solutions to improve project management in a distributed setting. [3]

4 Performing a Project in a Distributed Software Development Course: Lessons Learned

4.1 Author

Ciccozzi, F. and Crnkovic, I.

4.2 abstract

Distributed software development approaches have to face with several issues like cultural differences, collaboration and communication mechanisms, which can undermine the overall development success if not handled in a proper manner. In order to provide a real environment for students placed in different countries to learn and apply the best practices in distributed software development, a course has been developed jointly by two european universities. The course aims at providing the students an insight in the complexity of distributed development and giving the possibility to work in distributed teams for actual implementations, in order to minimize the gap between theory and practice. This paper describes the course design, challenges, results and success factors, from a students perspective. [1]

5 Challenges of Globally Distributed Software Development; Analysis of Problems Related to Social Processes and Group Relations

5.1 Author

Piri, A.

5.2 abstract

When software development is understood as collaborative action between group members, many of the common problems encountered in software development projects can be traced back to social factors of the project. In distributed projects, physical distance between group members limits their face-to-face interaction and thus sets special challenges to communication, which is essential for creating good relations between group members. The goal of this qualitative study is to develop understanding of the importance of social processes and group relations in globally distributed software development, by utilizing literature and earlier research both from software engineering and social sciences. [14]

6 Task Coordination in an Agile Distributed Software Development Environment

6.1 Author

David K.M. Mak and Philippe B. Kruchten

6.2 abstract

As both distributed software development (DSD) and agile development practices become more popular, the problem of task coordination in an agile DSD environment becomes more pertinent. Even though task allocation has been a subject of study for many years, the team dynamics in an agile DSD environment makes the nature of task coordination distinctly different from that in other disciplines. This paper proposes a solution to the problem of remote task allocation and coordination in an agile DSD environment. It combines current practices in software project management, such as object-oriented process modeling and critical-path analysis, and methodologies from other fields, such as workflow management and management science. It also describes NextMove, a Java/Eclipse-based distributed tool that would assist project managers in making day-to-day task allocation decisions, increasing transparency throughout the project, as well as complementing other modes of communication in a DSD environment. [8]

7 Applying Agile Principles for Distributed Software Development

7.1 Author

Phalnikar, R. and Deshpande, V.S. and Joshi, S.D.

7.2 abstract

The necessity of finding right skilled people, sharing resource and limitation on cost has made distributed software development indispensable. In a distributed development project, but are working collaboratively toward the outcome. Such offshore service providers follow the traditional process models. Agile practices promote development iterations, open collaboration, and process adaptability throughout the life cycle of the project. Adapting these practices in a distributed environment can help distributed development tackle the challenges of cultural incompatibility, leadership struggle and lack of trust. This paper describes the benefits of using agile process and Scrum the iterative incremental process in distributed software development, and proposes two team structures for its implementation. [13]

8 Usage of SCRUM Practices within a Global Company

8.1 Author

Cristal, M. and Wildt, D. and Prikladnicki, R.

8.2 abstract

Global companies that experimented extensive waterfall phased plans are trying to improve their existing processes to expedite team engagement. SCRUM has become an acceptable path to follow for those companies because it comprises project management as part of its practices. SCRUM has been used with the objective of simplifying project control through simple processes, easy to update documentation and higher team iteration over exhaustive documentation. Instead of investing team effort on producing static documentation, SCRUM proposes to focus on team continuous improvement aiming to add value to business processes. The purpose of this industry report is to describe two projects that experimented SCRUM practices within a globally distributed company. This company has development centers across North America, South America and Asia. This report covers challenges faced by the project teams, strengths and practical recommendations of using SCRUM in a globally distributed environment. [2]

9 Patterns of Evolution in the Practice of Distributed Software Development in Wholly Owned Subsidiaries: A Preliminary Capability Model

9.1 Author

Prikladnicki, R. and Damian, D. and Audy, J.

9.2 abstract

In this paper, we describe a preliminary capability model that captures patterns of evolution in the practice of distributed software development in internal offshoring projects. In our research we seek to understand how the practices of organizations involved in the internal offshoring of software development evolve over time, from a software engineering perspective, and from the point of view of the subsidiaries. Based on a combination of qualitative and quantitative methods, we propose a capability model that encompasses the evolution of software development activities within and among several subsidiaries owned by an organization. [15]

10 Technologies and Tools for Distributed Teams

10.1 Author

Rodriguez, J.P. and Ebert, C. and Vizcaino, A.

10.2 abstract

Software development today is typically a team effort with team members in different geographical places. You might regret that teamwork isn't what it used to be, but you also might want to look toward technologies and tools that support distributed teamwork. Authors Javier Portillo Rodriguez, Aurora Vizcaino, and I provide an overview of such technologies with many tools examples, starting with technologies such as Jazz and then showing how users can orchestrate individual tools in a distributed context. [16]

11 A New Perspective on GDSD Risk Management: Agile Risk Management

11.1 Author

Mudumba, V. and One-Ki, Lee

11.2 abstract

Risk management in globally distributed software development (GDSD) projects is becoming a critical area of concern for practitioners. The risks in GDSD projects can be dynamic due to the multiplicity in various aspects of GDSD projects (e.g., multi-locations, multi-cultures, multi-groups, multi-standards, and multi-technologies). This multiplicity nature leads to dynamic interactions among the internal (i.e., people, process, and technology) and external elements of a GDSD project. This study aims to develop a new framework to identify the dynamic risks in GDSD projects and mitigate them using agile risk management practices. We reflect the proposed framework on a case of GDSD project in the literature, which experienced high multiplicity and thus high dynamics in its project management. [10]

12 Identification of Success and Failure Factors of Two Agile Software Development Teams in an Open Source Organization

12.1 Author

Tsirakidis, P. and Kobler, F. and Krcmar, H.

12.2 abstract

Agile software development methods and free/libre open source development have been two embracing movements in the software industry for more than a decade. However, little is known about the composition of both while today a variety of studies provide us with key characteristics of each one. The study extracts the similarities and differences of both topics by means of an extensive literature review focusing on teams as research unit. Furthermore this study investigates two agile software development teams in an open source organization based on an explorative research design. The focus is on the empirical identification of success and failure in the application of agile methods in teams with open source background, structure and characteristics and in comparison with the literature findings. This study is still in progress, however first results as well as the methodology will be presented hereafter. [18]

13 Agile, open source, distributed, and on-time - inside the Eclipse development process

13.1 Author

Gamma, E.

13.2 abstract

Summary form only given. Eclipse is a widely recognized open source project dedicated to providing a platform for developing integrated tools. Throughout the history of Eclipse the development team was successful in hitting projected delivery dates with precision and quality. This isn't possible without a team strongly committed to ship quality software. How is this really done? How does Eclipse achieve quality and just-in-time delivery? This paper sheds light on the key practices of the Eclipse development process - from the development mantras "always beta", "milestones first", "API first", and "performance first" to practices such as ensuring quality through multiple feedback loops. The author reflects on proven practices for managing a large project performed by geographically dispersed teams and open source contributors in a highly competitive market. Most of these practices have evolved in the open source project, but they are equally applicable to closed source projects and help to improve quality, timeliness and reduce development stress in both types of environments. [5]

14 Trouble in paradise: the open source project PyPy, EU-funding and agile practices

14.1 Author

During, B.

14.2 abstract

PyPy is an open source project, partly funded by the European Union, employing agile techniques evolved within the Python Community such as "sprint-driven development". The project started as a grass-root F/OSS effort in late 2002 and received EU-funding from December 2004 until November 2006. In this paper we present the various influencing factors that creates the hybrid project process that is PyPy. These influencing factors are the F/OSS Python Community (climate of the community from which PyPy grew from), agile practices (such as the Python community evolved technique of "sprinting") and the EU funding practices (resource tracking and reporting) impacting the project. These influencing factors laid the foundation for the custom-made project process that makes this unique hybrid project work. The main factor for driving this process is the skills of the team of core developers instigating the project. PyPy, with its open and transparent communication and collaborative work style, is again a proof that the best agile practice is the people factor. [4]

15 A practical measure for the agility of software development processes

15.1 Author

Shawky, D.M. and Ali, A.F.

15.2 abstract

In the software industry, a large number of projects fail and billions of dollars are spent on failed software projects. Lack of an end user involvement, poor requirements, and unrealistic schedules are some of the top reasons of such failure. Agile software development is an approach that addresses these problems through a real communication between programmers and customers. Thus, there is a need to quantify software agility. In this paper, an approach for quantifying software agility is provided by modeling the key concepts in agile software and proposing a measure that can be used in representing how agile a software development process is. The proposed measure employs information entropy as the main concept related to software agility. The suggested measure is tested on two open source case studies. Experimental results demonstrate the validity and suitability of the agility measure. [17]

16 Enhancing the Performance of Software Development Virtual Teams through the Use of Agile Methods: A Pilot Study

16.1 Author

Nevo, S. and Chengalur-Smith, I.

16.2 abstract

This paper develops a conceptual model that explicates the role of synchronous communication media in enabling - directly and indirectly, via social presence - virtual software development teams to adopt and apply Agile methods. In turn, Agile methods, as well as perceived social presence, are theorized to have a positive impact on communication convergence and transactive memory. Ultimately, these outcomes are formulated as direct antecedents of virtual team performance. A pilot study of 40 Free/Libre Open Source Software (FLOSS) teams provides preliminary supporting evidence for the conceptual model. [11]

17 The Creation of a Distributed Agile Team

17.1 Author

Karsten, Paul and Cannizzo, Fabrizio

17.2 abstract

This report tells the story of a project started one and a half years ago in BT and how the enthusiasm and dedication on applying agile methodologies has allowed the team to grow while successfully delivering on their goals. It describes the process that has been put in place to manage the project and develop the software; it also tells how some of the practices initially applied have been then changed and adapted to make them fit for the distributed and unique nature of the team. [6]

18 Analysis of Emergent and Evolving Information: The Agile Planning Case

18.1 Author

Petersen, Rasmus Rosenqvist and Wiil, Uffe Kock

18.2 abstract

Some information structures are by nature emergent and evolving and as a consequence the retrievable knowledge keeps shifting patterns like a kaleidoscope. Hence, information analysis can be a complex and tedious task. The planning task of agile teams is an example of such a complex information analysis task. In this paper, we present a lightweight planning tool. ASAP is inspired by concepts and principles from spatial hypertext, which have proven successful in supporting information analysis tasks. ASAP runs on a large interactive vertical display on which electronic task cards can be organized into iterations and releases using card hierarchies and separators (a novel visual concept). Several views of the evolving plan are automatically generated to assist the agile team with overviews of tasks, estimates, and assignments. Views are instantly updated to reflect changes to the plan. [12]

19 Using Horizontal Displays for Distributed and Collocated Agile Planning

19.1 Author

Morgan, Robert and Walny, Jagoda and Kolenda, Henning and Ginez, Estaban and Maurer, Frank

19.2 abstract

Computer-supported environments for agile project planning are often limited by the capability of the hardware to support collaborative work. We present DAP, a tool developed to aid distributed and collocated teams in agile planning meetings. Designed with a multi-client architecture, it works on standard desktop computers and digital tables. Using digital tables, DAP emulates index card based planning without requiring team members to be in the same room. [9]

20 Evaluating the Effect of Agile Methods on Software Defect Data and Defect Reporting Practices - A Case Study

20.1 Author

Korhonen, K.

20.2 abstract

In large, traditional software development projects, the number of defects can be considerably high. Agile methods promise code quality improvement, but while embracing the agile methods, software development organizations have realized that defects still do exist and must be managed. When the development is distributed over several sites, defect management can become even more challenging. In this study we analyzed defect data in a large multi-site organization during the first twelve months of their agile transformation. Complementing information was gathered by a survey, which was conducted in the organization twice: after six and after twelve months of starting the agile transformation. The results indicate that the defect reporting practices changed after the agile adoption was started, the defect inflow was more stable and the defect closing speed improved. [7]

References

1. F. Ciccozzi and I. Crnkovic. Performing a project in a distributed software development course: Lessons learned. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 187 –191, aug 2010.
2. M. Cristal, D. Wildt, and R. Prikladnicki. Usage of scrum practices within a global company. In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, pages 222 –226, aug 2008.
3. F.Q.B. da Silva, C. Costa, A.C.C. Frana, and R. Prikladinicki. Challenges and solutions in distributed software development project management: A systematic literature review. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 87 –96, aug 2010.
4. B. During. Trouble in paradise: the open source project pypy, eu-funding and agile practices. In *Agile Conference, 2006*, pages 11 pp. –231, july 2006.
5. E. Gamma. Agile, open source, distributed, and on-time - inside the eclipse development process. In *Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on*, page 4, may 2005.
6. Paul Karsten and Fabrizio Cannizzo. The creation of a distributed agile team. In Giulio Concas, Ernesto Damiani, Marco Scotto, and Giancarlo Succi, editors, *Agile Processes in Software Engineering and Extreme Programming*, volume 4536 of *Lecture Notes in Computer Science*, pages 235–239. Springer Berlin / Heidelberg, 2007.
7. K. Korhonen. Evaluating the effect of agile methods on software defect data and defect reporting practices - a case study. In *Quality of Information and Communications Technology (QUATIC), 2010 Seventh International Conference on the*, pages 35 –43, oct. 2010.
8. David K.M. Mak and Philippe B. Kruchten. Task coordination in an agile distributed software development environment. In *Electrical and Computer Engineering, 2006. CCECE '06. Canadian Conference on*, pages 606 –611, may 2006.
9. Robert Morgan, Jagoda Walny, Henning Kolenda, Estaban Ginez, and Frank Maurer. Using horizontal displays for distributed and collocated agile planning. In Giulio Concas, Ernesto Damiani, Marco Scotto, and Giancarlo Succi, editors, *Agile Processes in Software Engineering and Extreme Programming*, volume 4536 of *Lecture Notes in Computer Science*, pages 38–45. Springer Berlin / Heidelberg, 2007.
10. V. Mudumba and Lee One-Ki. A new perspective on gdsd risk management: Agile risk management. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 219–227, aug 2010.
11. S. Nevo and I. Chengalur-Smith. Enhancing the performance of software development virtual teams through the use of agile methods: A pilot study. In *System Sciences (HICSS), 2011 44th Hawaii International Conference on*, pages 1 –10, jan. 2011.
12. Rasmus Rosenqvist Petersen and Uffe Kock Wiil. Analysis of emergent and evolving information: The agile planning case. In Jos Cordeiro, AlpeshKumar Ranchordas, and Boris Shishkov, editors, *Software and Data Technologies*, volume 50 of *Communications in Computer and Information Science*, pages 263–276. Springer Berlin Heidelberg, 2011.
13. R. Phalnikar, V.S. Deshpande, and S.D. Joshi. Applying agile principles for distributed software development. In *Advanced Computer Control, 2009. ICACC '09. International Conference on*, pages 535 –539, jan. 2009.

14. A. Piri. Challenges of globally distributed software development; analysis of problems related to social processes and group relations. In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, pages 264 –268, aug 2008.
15. R. Prikladnicki, D. Damian, and J. Audy. Patterns of evolution in the practice of distributed software development in wholly owned subsidiaries: A preliminary capability model. In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, pages 99 –108, aug 2008.
16. J.P. Rodriguez, C. Ebert, and A. Vizcaino. Technologies and tools for distributed teams. *Software, IEEE*, 27(5):10–14, sept.-oct. 2010.
17. D.M. Shawky and A.F. Ali. A practical measure for the agility of software development processes. In *Computer Technology and Development (ICCTD), 2010 2nd International Conference on*, pages 230 –234, nov. 2010.
18. P. Tsirakidis, F. Kobler, and H. Krcmar. Identification of success and failure factors of two agile software development teams in an open source organization. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 295 –296, july 2009.
19. S.M. White. Application of cognitive theories and knowledge management to requirements engineering. In *Systems Conference, 2010 4th Annual IEEE*, pages 137 –142, 2010.