# Chapter 1
# Fundamentals of Agile Distributed Software Development

**Darja Šmite, Nils Brede Moe, and Pär J. Ågerfalk**

**Abstract** This chapter provides an introduction to the area of agile distributed software development. It proceeds as follows. We start by introducing and motivating (globally) distributed software development, and follow on with agile software development. With this foundation we discuss the concept of agile distributed development, its motivation and some of the pertinent issues involved.

## 1.1 Introduction

### *1.1.1 Distributed Software Development*

In the current era of globalization, cross-national and cross-organizational collaboration has become a natural evolution in the operation of the global marketplace. Tight budgets, limited resources and time constraints have motivated many companies to explore global sourcing. This mode of working promises organizations the benefits of reaching mobility in resources, obtaining extra knowledge through recruiting the most talented people around the world, reducing time-to-market, increasing operational efficiency, improving quality, expanding through acquisitions, reaching proximity to market, and many more. As a result, a growing number of software companies have started to implement global supply chains.

D. Šmite (✉)
Blekinge Institute of Technology, Ronneby, Sweden
e-mail: darja.smite@bth.se

N.B. Moe
SINTEF ICT, Trondheim, Norway
e-mail: nilsm@sintef.no

P.J. Ågerfalk
Uppsala University, Uppsala, Sweden
e-mail: par.agerfalk@im.uu.se

D. Šmite et al. (eds.), *Agility Across Time and Space*,
DOI 10.1007/978-3-642-12442-6_1, 

While companies are taking the assumed benefits almost for granted, industrial experience shows that these are not as easy to achieve as the literature may lead one to believe [1]. In contrast to other engineering disciplines, developing software is recognized as a significantly complex task that heavily relies on human interaction. Accordingly, distributed software projects with geographically, temporally and socio-culturally dispersed teams unavoidably experience unique pressures and challenges. There are major problems related to communication, coordination and collaboration caused by geographical, temporal and socio-cultural distance.

### *1.1.2 Agile Software Development*

Setting up an agile team is usually motivated by benefits such as increased productivity, innovation, and employee satisfaction. However the agile approach is also motivated by the increasing complexity of software development. As information technology's role in the modern economy has grown in importance, software developers have found themselves confronted with the challenges of exceptional complexity. A software team needs to interact with and consider the viewpoints of a wide variety of stakeholders, many of whom have conflicting views on the software features and functionality. The team must then balance disparate needs of diverse stakeholders, a task far more challenging than merely fulfilling the functional requirements of a system. Thus, it comes as no surprise that many software developers more than welcome agile software development, which embraces these emerging realities at its core.

The agile approach is built around empowered and self-organizing teams that coordinate their work themselves. In such teams there is also a strong focus on collaboration and communication. Collaboration and coordination depend on communication, which is central to successful software development. These activities are supported through various agile practices including pairing, customer collaboration, stand-ups, reviews, retrospectives and the planning game.

## 1.2 Merging Agility with Distribution

Addressing the problems related to the complexity of software development and the strong focus on collaboration, coordination and communication, are primary reasons for the growing interest in exploring the applicability of agile approaches in distributed software development. Despite the popularity of the topic, the practice of agile development has been well ahead of research in the field [2]. Thus, there is still no consensus or deep, theoretically grounded, understanding of the applicability of agile methods to different types of software projects and the flexibility in application of agile methods necessary to realise the benefits promised.

**Table 1.1** Characteristics of agile versus traditional distributed software development

| Characteristics | Agile Development | Distributed Development |
|---|---|---|
| Communication | Informal | Formal |
| | Face-to-face | Computer-mediated |
| | Synchronous | Often asynchronous |
| | Many-to-many | Tunneled |
| Coordination | Change-driven | Plan-driven |
| | Mutual adjustment, self-management | Standardization |
| Control | Lightweight | Command-and-control |
| | Cross-functional team | Clear separation of roles |

### *1.2.1 Potential Issues*

While the motivation for implementing agility in distributed software development is clear, the process of marrying agility with distribution is not straightforward. Looking at the principles of agile development and the environment of distributed projects, one can easily characterize the two as opposite extremes on a continuum. The fundamental differences between agile and distributed development can be illustrated by the following examples (see Table 1.1).

Methodological standardization has often been argued to be the most effective way to manage global software teams [3]. Consequently, global managers tend to rely on plan-driven methods, in which the life cycle model specifies the tasks to be performed and the desired outcomes of each project phase. These projects are often characterized by defined task division, strict role separation, and preferably complete documentation. Managers are required to perform proper upfront planning and check adherence to the processes through supervision.

However, because of geographical, temporal and socio-cultural distance, standardization and command-and-control oriented management often fail. Distributed development is associated with computer-mediation, asynchronous communication, and lack of transparency for remote activities. Thus, the applicability of these coordination mechanisms is typically insufficient. This motivates the application of agile methods based on mutual adjustment and teamwork for distributed project coordination.

These differences between agile and distributed foundations suggest that the application of agile methods in distributed environments is doomed to fail. In fact, many believe that being agile and distributed is unrealistic. For example, Kontio et al. [4] claim that agile practices are hard to implement in distributed teams especially when the team size is large. On a similar note, Taylor, Greer et al. [5] claim that distributed agile software development suffers substantial difficulties because of its complex development environment and lack of empirical evidence describing the actual development experiences.

Due to geographical separation of stakeholders and teams in distributed projects, many of the fundamental concepts promoted by agile approaches are indeed difficult

to apply. The implementation of such practices as pair-programming, shared code ownership and onsite customer, puts demands on the distributed projects and leads to tailoring the practices and compensating the lack of co-location and face-to-face interaction through innovative information technology and communication tools. However, it has been argued that there are issues that cannot be solved through tools. For example, it is widely believed that trust needs touch. Accordingly, the application of agile methods for distributed projects, and thus the extent to which the benefits of agility can be achieved, may be hampered.

### *1.2.2 All or Nothing versus Á la carte*

While agile methods promote flexibility, some agilists advocate an all-or-nothing attitude to the selection and application of the methods and practices. This is because only the synergetic combination is recognized to guarantee the maximum benefits. Hence, the tailoring attitude to agile methods is often not very well received. The question of the viability of agile approaches in distributed projects then certainly springs to mind. If distributed projects are forced to select and tailor agile methods and practices, therefore following á la carte approach, what are the benefits that one can expect? With this aim a series of studies have been conducted and experiences clearly show that agility across time and space not only exists but also gains success. As long as developers understand the rationale behind certain practices suggested by a method, tailoring or even replacing parts should not be a problem. The issue at stake is rather to make sure that this rationale is communicated to developers for them to act upon in an informed way.

## 1.3 Current Practice

Previous work and published experiences (e.g. by some of the authors and editors of this book) show successful implementation of agile values and principles in different distributed projects. This motivates the assessment of the viability of agile practices for distributed software development teams. Working on this book we have sought to understand the major areas of interest covered by current research. With this aim, we conducted a literature study and identified 41 relevant research papers from the following conferences: XP, ICGSE, Agile, Euromicro, HICSS, COMPSAC, ASPEC and EuroSPI. Research topics covered by these articles included:

- Benefits of introducing agility in distributed projects;
- Adopting agility for distributed projects;
- Communication in agile distributed projects;
- Planning and coordination in agile distributed projects;
- Customer relationship in agile distributed projects;
- Tool support for implementing agility in distributed projects;

- Recommendations for implementing agility in distributed projects;
- XP in distributed projects;
- Scrum in distributed projects;
- User stories in distributed projects.

These topics illustrate three trends. First, researchers are exploring the benefits of agile methods in distributed environment. Second, considerable effort is put into exploring the practices that unavoidably require tailoring through, for example, tool support. Finally, a large number of articles are dedicated to investigating the best practices for implementing agility, including attempts of validation of selected agile methods (or parts thereof) in distributed environments.

## 1.4 Conclusions

Arguably, the combination of agile and distributed development is of immense interest to industry. Agile development practice has always been ahead of research, with academics struggling to understand what is going on and why it apparently works so well. The same is true about agile distributed development, where practitioners started to experiment and quickly adjust their strategies. As a result, a number of agile methods have been tried out in distributed projects, and there is certainly a lot to be learnt from that experience.

Despite the popularity of the topic, we still do not understand fully the limitations and viability of agile methods in seemingly incompatible environments of distributed software projects. Agile methods work well in the settings they were designed for (i.e. small co-located teams). How they will play out in large, globally distributed projects, is still an open question. However, the remainder of this book is devoted to explore this question and provide actionable advice for anyone embarking on such a quest, or is just interested in the area for any other reason (including being a researcher, perhaps).

## References

1. Ó Conchúir, E., Ågerfalk, P. J., Fitzgerald, B., & Holmström Olsson, H. (2009). Global software development: Where are the Benefits? *Communications of the ACM*, *52*(8), 127–131.
2. Ågerfalk, P. J., & Fitzgerald, B. (2006). Flexible and distributed software processes: Old petunias in new bowls? *Communications of the ACM*, *49*(10), 26–34.
3. Carmel, E. (1999). *Global software teams: Collaborating across borders and time zones*. Englewood Cliffs: Prentice-Hall.
4. Kontio, J., Hoglund, M., Ryden, J., & Abrahamsson, P. (2004). Managing commitments and risks: challenges in distributed agile development. In *Proceedings of the international conference on software engineering* (pp. 732–733).
5. Taylor, P. S., Greer, D., Sage, P., Coleman, G., McDaid, K., & Keenan, F. (2006). Do agile GSD experience reports help the practitioner? In *Proceedings of the 2006 international workshop on global software development of ACM* (pp. 87–93).