Information and Software Technology 64 (2015) 132–147

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments

Fabian Fagerholm [a,*], Marko Ikonen [a], Petri Kettunen [a], Jürgen Münch [a], Virpi Roto [b], Pekka Abrahamsson [c]

[a] Department of Computer Science, University of Helsinki, P.O. Box 68, FI-00014 University of Helsinki, Finland
[b] School of Arts, Design and Architecture, Aalto University, P.O. Box 31000, 00076 Aalto University, Finland
[c] Faculty of Computer Science, Free University of Bozen-Bolzano, Piazza Domenicani 3, Bolzano, Italy

A R T I C L E I N F O

Article history:
Received 1 September 2014
Received in revised form 30 December 2014
Accepted 20 January 2015
Available online 2 February 2015

Keywords:
Developer Experience
Team performance
Case study
Human factors
Agile software development
Lean software development

A B S T R A C T

Context: Companies increasingly strive to adapt to market and ecosystem changes in real time. Gauging and understanding team performance in such changing environments present a major challenge.
Objective: This paper aims to understand how software developers experience the continuous adaptation of performance in a modern, highly volatile environment using Lean and Agile software development methodology. This understanding can be used as a basis for guiding formation and maintenance of high-performing teams, to inform performance improvement initiatives, and to improve working conditions for software developers.
Method: A qualitative multiple-case study using thematic interviews was conducted with 16 experienced practitioners in five organisations.
Results: We generated a grounded theory, Performance Alignment Work, showing how software developers experience performance. We found 33 major categories of performance factors and relationships between the factors. A cross-case comparison revealed similarities and differences between different kinds and different sizes of organisations.
Conclusions: Based on our study, software teams are engaged in a constant cycle of interpreting their own performance and negotiating its alignment with other stakeholders. While differences across organisational sizes exist, a common set of performance experiences is present despite differences in context variables. Enhancing performance experiences requires integration of soft factors, such as communication, team spirit, team identity, and values, into the overall development process. Our findings suggest a view of software development and software team performance that centres around behavioural and social sciences.

1. Introduction

Performance is a multi-faceted concept that is used on several levels of an organisation to mean different things [30]. The desired outcome, a successful and well-performing software product or service, is contingent on a complex combination of factors that can be found in projects, processes, organisations, teams, and individuals (e.g. [54,50,58,59]). Within these categories, there are multiple characterisations of performance that are relevant in different contexts and for different purposes. Even the performance of the end result, the software itself, can be viewed in different ways; e.g. in terms of technical quality, fitness for purpose, or generated profits. Many of today's software development organisations operate in highly volatile environments in which different elements of performance can change rapidly. As corporate strategy changes, performance targets may sometimes change implicitly, sliding continuously to meet the updated understanding of conditions in the business milieu. Some organisations aim to improve performance by being more responsive to changing market needs, e.g. by treating R&D as a continuous experimentation system [40]. However, propagating goal changes to all levels of the organisation in a comprehensive and timely manner may be hampered by communication and transparency problems. Also, if goals change too

* Corresponding author.
E-mail addresses: fabian.fagerholm@helsinki.fi (F. Fagerholm), marko.ikonen@alumni.helsinki.fi (M. Ikonen), petri.kettunen@cs.helsinki.fi (P. Kettunen), juergen.muench@cs.helsinki.fi (J. Münch), virpi.roto@aalto.fi (V. Roto), pekka.abrahamsson@unibz.it (P. Abrahamsson).

http://dx.doi.org/10.1016/j.infsof.2015.01.010
0950-5849/© 2015 Elsevier B.V. All rights reserved.

quickly and frequently, organisational activity may become erratic and self-defeating.

When the objective is to analyse and understand teams, human factors are brought to the forefront. A team may be evaluated, e.g., in terms of its productivity [59], speed [7], or ability to produce novel and innovative results [45]. It may also be evaluated in terms of process control [54], or the knowledge it produces [55]. Many factors influence performance within these areas and time is frequently an important factor to consider. However, since software development is largely a human-based activity, most types of outcome depend on human factors. Motivation, skill, satisfaction, values, and personality are factors to consider when forming teams, creating and designing processes and development environments, and structuring organisations and communication. The importance of such human aspects on performance in software development is well known [6,8,26,37,59]. However, there is a lack of understanding in many software development environments of how software practitioners themselves experience the pursuit of high performance, and how striving for performance could simultaneously be a meaningful and positive experience.

In a previous paper [27], we studied how professional software developers experience performance in a Lean and Agile context. Drawing on an earlier conceptualisation of Developer Experience [28], we approached the issue through a cognitive, affective, and conative lens. We viewed team performance from the perspective of individual software practitioners, gaining insights that may be of use in evaluating teams from an internal perspective. The study showed why it is not sufficient to consider performance only as meeting predefined objectives. It also showed how practitioners reason as they attempt to perform in their work, and what they perceive as beneficial and detrimental for those attempts.

The present article is an extension of the previous study that adds additional analysis. We aim to cast further light on the similarities and differences in performance experiences among professional software developers in different types of companies. We augment our previous results with findings that show reasoning appearing consistently across companies of different types, and reasoning that emerges when moving between types: from smaller to larger companies, between companies in different fields of industry, and different degrees of globalisation. We also show that understanding how individual software developers experience the striving for performance in their teams can help formulate hypotheses of how and why the company is currently performing in its software development activities. Such hypotheses may be of use in performance improvement efforts, such as software process improvement initiatives. Our specific research questions are:

RQ1 How do software practitioners experience team performance in Lean and Agile environments?
RQ2 How do software practitioners reason about the relationships between perceived performance factors?
RQ3 How do performance factors experienced by software practitioners differ between different types of companies?

The remainder of the article is organised as follows. In Section 2, we discuss the concept of performance in software engineering, with particular focus on human factors on the team and individual levels. In Section 3, we describe our research approach: the data collection and analysis methods used. In Section 4, we present the empirical results. We discuss the implications and limitations of our findings in Section 5. Finally, we conclude the paper in Section 6 and briefly outline possible future work.

2. Theoretical background

One of the foremost practical objectives of team performance research is the pursuit of ways to improve the work outcome of teams. It is interesting to note that teams were once considered an improvement over individual work: teams can potentially offer greater adaptability, productivity, and creativity than any single individual [31,35,61]. However, gaining the potential benefits of teams is not easy. For example, it is not enough to merely group skilled individuals together [36]. In this section, we briefly discuss how to define performance, and shortly review some previous research on performance factors and models of team performance.

2.1. Definition of performance

One definition of high-performing teams is that they outperform “all reasonable expectations as well as all other similarly situated teams” [43]. While this definition proceeds to say that the performance of these teams surprises even themselves, organisations find high-performing teams highly desirable and wish to replicate their success. However, reports describing such high-performing teams are typically on an anecdotal level, based more on assumptions than on a valid causal analysis. Part of the problem may stem from the lack of a sound measure for “success” in software engineering, although it is a central dependent variable [57].

Performance is often divided into efficiency and effectiveness. Efficiency means accomplishing objectives quickly and with minimal resource usage. Effectiveness refers to accomplishing the right objectives, e.g. those that have the greatest value. However, the terms can be used differently; e.g. Salas et al. [61] use them as follows. Team performance refers to “the outcomes of the team’s actions regardless of how the team may have accomplished the task”. Team effectiveness considers “not only whether the team performed” (e.g. completed a task), but also “how the team interacted to achieve the team outcome” (e.g. team processes, teamwork). The distinction is important since many factors may influence the outcome, and confound the causal reasoning assumed in team performance measures. This may result in an incorrect understanding of the team and the group processes which govern its performance [61]. In this work, we use “performance” as an umbrella term for all the meanings described above and use more specific terms as needed.

2.2. Performance influence factors

Sudhakar et al. [65] list four classes of factors which influence team performance: (i) technical, (ii) non-technical (soft), (iii) organisational, and (iv) environmental. The technical factors include project-specific traits such as size, complexity, and processes, as well as product characteristics. There are numerous reported soft factors, and fully explaining them is beyond the scope of this paper. However, some examples can be mentioned.

On the individual level, cognitive factors include skill [9,10,65], knowledge [49], competence [37], and logical reasoning [13]. Motivation is a conative factor that has received much attention in software engineering research [8,29]. Personal values [49], beliefs [23,56], and personality [8,65] have also been investigated as direct or indirect performance factors. In addition, affective factors have been examined, showing that developers do experience several emotions in their work, and that these change over time [64]. Moods can influence programming tasks such as debugging [46]. Enthusiasm [67], and emotional valence and dominance [34], can have a positive effect on performance, while frustration is a negative risk factor for performance [67].