Information and Software Technology xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Information and Software Technology

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

Using simulation for understanding and reproducing distributed software development processes in the cloud

M. Ilaria Lunesu[*,a], Jürgen Münch[b], Michele Marchesi[c], Marco Kuhrmann[d]

[a] Department of Electrical and Electronic Engineering, University of Cagliari, Italy
[b] Herman Hollerith Center, Böblingen & Reutlingen University, Germany
[c] Department of Mathematics and Computer Science University of Cagliari, Italy
[d] Clausthal University of Technology, Institute for Applied Software Systems Engineering, Germany

ARTICLE INFO

Keywords:
Scrum
Kanban
Process simulation
Comparison

ABSTRACT

Context: Organizations increasingly develop software in a distributed manner. The Cloud provides an environment to create and maintain software-based products and services. Currently, it is unknown which software processes are suited for Cloud-based development and what their effects in specific contexts are.

Objective: We aim at better understanding the software process applied to distributed software development using the Cloud as development environment. We further aim at providing an instrument, which helps project managers comparing different solution approaches and to adapt team processes to improve future project activities and outcomes.

Method: We provide a simulation model, which helps analyzing different project parameters and their impact on projects performed in the Cloud. To evaluate the simulation model, we conduct different analyses using a Scrumban process and data from a project executed in Finland and Spain. An extra adaptation of the simulation model for Scrum and Kanban was used to evaluate the suitability of the simulation model to cover further process models.

Results: A comparison of the real project data with the results obtained from the different simulation runs shows the simulation producing results close to the real data, and we could successfully replicate a distributed software project. Furthermore, we could show that the simulation model is suitable to address further process models.

Conclusion: The simulator helps reproducing activities, developers, and events in the project, and it helps analyzing potential tradeoffs, e.g., regarding throughput, total time, project size, team size and work-in-progress limits. Furthermore, the simulation model supports project managers selecting the most suitable planning alternative thus supporting decision-making processes.

1. Introduction

Being able to collaborate effectively has become a crucial factor in software development and maintenance. Organizations increasingly develop software in a distributed manner by appointing external developers and development teams, who collaboratively work at different sites utilizing a multitude of communication tools [9,36]. Literature shows distributed software development being challenged by many factors, e.g., distance in language, culture, time and location, coordination of distributed (virtual) teams, and lack of trust among developers [18,40]. Notably agile software development constitutes a challenge, as agile software development relies on a set of principles and values that put the people and close collaboration and interaction in the spotlight. It is crucial to understand how agile methods "behave" in distributed software development as adapting and deploying an agile method to a project spanning several sites bears some risk [28].

A simulation-based approach grounded in statistical data from previous projects can help analyzing risks and evaluating different process variants [23,52], but also helps evaluating decisions and potential effects on a project [8]. Moreover, a process simulation offers insights faster than a full case study [14, pp. 11–13]. In particular, a *simulation model* can be modified and the results quickly provide indication whether or not modified parameters affect a project and how—so-called "what-if" analyses [56]. For example, while it is hard to modify the team in a "real" project, in a simulation, modifying the *team size* parameter helps analyzing the impact, e.g., on work-in-progress

* Corresponding author.
E-mail address: ilaria.lunesu@diee.unica.it (M.I. Lunesu).

https://doi.org/10.1016/j.infsof.2018.07.004
Received 28 January 2018; Received in revised form 2 July 2018; Accepted 3 July 2018
0950-5849/ © 2018 Elsevier B.V. All rights reserved.

(WIP), lead/cycle time, and team productivity. Furthermore, a simulation model provides flexibility to allow for configuring different process models, running simulations on a shared dataset, and to compare and study aspects of interest of different process models. For instance, project managers interested in minimizing cycle times can use a simulation to compare the behavior of Scrum- and Kanban-based processes to pick the process variant promising the best performance. In this regard, a simulation can be utilized to modify parameters, find relations between parameters, and study complex processes over time. According to Kellner et al. [23] and Armbrust et al. [8], a simulation used this way can help reproducing a real system, compare variants, identify bottlenecks, and so forth. Hence, a process simulation is a tool to help project managers analyzing different actions, evaluating impact, and eventually selecting those actions best fitting a particular situation [29].

1.1. Problem statement

Even though globally distributed software development (also called Global Software Development; GSD, or *Global Software Engineering; GSE*) is around for years, still, practitioners struggle with effectively adapting agile methods [28]. In this context, the *Cloud* provides a highly flexible environment offering a variety of services. However, little is known which processes are used for distributed development using the *Cloud as software development environment*, how these processes are used and customized, and how they might differ from other approaches.

1.2. Objective

Our overall objective is to better understand the software process applied in GSE settings, notably settings using the Cloud as development environment. Based on real project data,[1] a simulation-based approach was chosen to improve the understanding of such processes and to support project managers to select and tailor software processes for Cloud-based distributed software development. Hence, an objective of the presented work is also to show feasibility/reliability of using simulation models, e.g., for projects in the *Software Factory* environment. Finally, we aim at providing an instrument, which helps project managers comparing different solution approaches and to adapt current team processes to improve future project activities and outcomes.

1.3. Contribution

An event-driven simulator [7] was configured using a Scrumban process with the number of user stories and their effort and priority in the backlog as input. The simulator helps reproducing activities, developers, user stories and events in the project, and it generates statistics, e.g., on throughput, total time, and lead and cycle time. The resulting simulation model can be customized to simulate different processes. Specifically, in addition to the Scrumban process, we also modeled "pure" Scrum and Kanban processes to allow for comparing the different processes with regard to project performance thus supporting project managers in selecting the best-fitting development approach for a specific scenario.

1.4. Outline

The remainder of the article is organized as follows: Section 2 provides an overview of related work. In Section 3, we describe the research design including research questions, simulation variables, and the specification and implementation of the simulation model. Section 4 presents the results from the different simulations. We conclude this article in Section 5.

2. Related work

2.1. Software Processes and GSE

Globally distributed software development has become commodity, and it was showcased that distributed teams and even outsourced teams can be as productive as small collocated teams [43], which, however, requires a full implementation of Scrum along with good engineering practices. Paasivaara et al. [34] state that agile methods can provide a competitive advantage by delivering early, simplifying communication and allowing the business to respond more quickly to the market by changing the software. To support this claim, authors present a multi-case study on the application of Scrum practices to three globally distributed projects discussing challenges and benefits. In this regard, Phalnikar et al. [35] propose two team structures for implementing Scrum in a distributed setting. However, deploying agile methods to a GSE-setting is challenging for several reasons, such as demanding communication in a distributed setup, challenges related to coordination, and collaboration [5,28,49], and there is yet no agreement on generalizable solution approaches. For instance, while Vallon et al. [49] discuss how agile practices can help improving or resolving such issues and found Scrum the most promising/successful development approach, Lous et al. [28] found GSE challenging Scrum, especially when it comes to scaling the process in the context of (large) distributed settings. Wang et al. [53] state that using agile methods helps mitigating challenges in co-located as well as in distributed teams, e.g., responding to fast-paced changes that occur in software projects. All the factors above influence the way in which software is defined, built, tested, and delivered. Ramesh et al. [37] discuss how to integrate and balance agile and distributed development approaches to address such typical challenges in distributed development.

Complementing the "pure" agile approaches, *Lean* approaches have gained significance in the software industry, and they are used in co-located and distributed settings alike. Such approaches focus on eliminating waste, e.g., [33], yet, these approaches are still under study, notably with regards to the question if and how these approaches help mitigating the various challenges in GSE. For instance, Tanner and Dauane [44] study Kanban and highlight those elements that can help alleviating communication and collaboration issues in GSE. Kanban is a development approach, which applies Lean principles [1,2,22] and is becoming increasingly popular as an effective extension of Scrum and other agile methods. However, even though Kanban's popularity is increasing, many questions regarding its adoption in software development remain open. Practitioners face serious challenges while implementing Kanban, since clear definitions of its practices, principles, techniques, and tools are missing. In response, distributed teams use a plethora of specific tools to facilitate collaborative work [36]. However, different studies suggest the projects' processes being selected in a pragmatic rather than in a systematic manner [25,45,50], and studies also suggest agile methods stepping into the background when it comes to define proper tool support [16]. On the other hand, GSE is a discipline that is maturing, as for instance Šmite et al. [51] show in their discussion of available empirical evidence in the field or Ebert et al. [13] who discuss the impact of GSE-related research to industry. That is, there is a variety of software processes and support tools used in practice. Such combinations are usually made in response to the respective project context [25], which gives project managers a hard time picking the most efficient process-tool combination for a project.

2.2. Software process simulation

Software Process Modeling Simulation (SPMS) is presented as a

[1] For seven weeks, six developers in Finland and six in Spain, located at three sites (two in Spain and one in Finland) worked on a project developing a SmartGrid system. See Section 4.1 for further details.

Download English Version:

https://daneshyari.com/en/article/8953937

Download Persian Version:

https://daneshyari.com/article/8953937

Daneshyari.com