The Journal of Systems and Software 000 (2014) 1–15

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

Collaboration optimization in software process composition

Andréa Magalhães Magdaleno [a,*], Marcio de Oliveira Barros [b,2], Cláudia Maria Lima Werner [a,1], Renata Mendes de Araujo [b,2], Carlos Freud Alves Batista [c,3]

[a] COPPE/UFRJ – Systems Engineering and Computer Science Program – Federal University of Rio de Janeiro (UFRJ), P.O. Box 68511, 21945-970 Rio de Janeiro – RJ, Brazil.
[b] Post-graduate Information Systems Program, Federal University of the State of Rio de Janeiro (PPGI/UNIRIO), Av. Pasteur 458, 22290-240 Urca – Rio de Janeiro – RJ, Brazil.
[c] Petrobras TIC/CPSW/PGOD/PMSW Rua da Assembleia, 100, 20011-904 Centro – Rio de Janeiro – RJ, Brazil.

ARTICLE INFO

Article history:
Received 30 November 2013
Revised 6 October 2014
Accepted 15 November 2014
Available online xxx

Keywords:
Collaboration
Software process
SBSE

ABSTRACT

Purpose: The purpose of this paper is to describe an optimization approach to maximize collaboration in software process composition. The research question is: how to compose a process for a specific software development project context aiming to maximize collaboration among team members? The optimization approach uses heuristic search algorithms to navigate the solution space and look for acceptable solutions.

Design/methodology/approach: The process composition approach was evaluated through an experimental study conducted in the context of a large oil company in Brazil. The objective was to evaluate the feasibility of composing processes for three software development projects. We have also compared genetic algorithm (GA) and hill climbing (HC) algorithms driving the optimization with a simple random search (RS) in order to determine which would be more effective in addressing the problem. In addition, human specialist point-of-view was explored to verify if the composed processes were in accordance with his/her expectations regarding size, complexity, diversity, and reasonable sequence of components.

Findings: The main findings indicate that GA is more effective (best results regarding the fitness function) than HC and RS in the search of solutions for collaboration optimization in software process composition for large instances. However, all algorithms are competitive for small instances and even brute force can be a feasible alternative in such a context. These SBSE results were complemented by the feedback given by specialist, indicating his satisfaction with the correctness, diversity, adherence to the project context, and support to the project manager during the decision making in process composition.

Research limitations: This work was evaluated in the context of a single company and used only three project instances. Due to confidentiality restrictions, the data describing these instances could not be disclosed to be used in other research works. The reduced size of the sample prevents generalization for other types of projects or different contexts.

Implications: This research is important for practitioners who are facing challenges to handle diversity in software process definition, since it proposes an approach based on context, reuse and process composition. It also contributes to research on collaboration by presenting a collaboration management solution (COMPOOTIM) that includes both an approach to introduce collaboration in organizations through software processes and a collaboration measurement strategy. From the standpoint of software engineering looking for collaborative solutions in distributed software development, free/open source software, agile, and ecosystems initiatives, the results also indicate how to increase collaboration in software development.

Originality/value: This work proposes a systematic strategy to manage collaboration in software development process composition. Moreover, it brings together a mix of computer-oriented and human-oriented studies on the search-based software engineering (SBSE) research area. Finally, this work expands the body of knowledge in SBSE to the field of software process which has not been properly explored by former research.

* Corresponding author. Tel.: +55 21 2562 8702; fax: +55 21 2562 8676.
E-mail addresses: andrea@cos.ufrj.br (A.M. Magdaleno), marcio.barros@uniriotec.br (M. de Oliveira Barros), werner@cos.ufrj.br (C.M.L. Werner), renata.araujo@uniriotec.br (R.M. de Araujo), carlos.freud@petrobras.com.br (C.F.A. Batista).

[1] Tel.: +55 21 2562 8702; fax: +55 21 2562 8676.
[2] Tel.: +55 21 2530 8088; fax: +55 21 2530 8051.
[3] Tel.: +55 21 3487 5213; fax: +55 21 3487 5213.

http://dx.doi.org/10.1016/j.jss.2014.11.036
0164-1212/

1. Introduction

Software development is a complex process that involves the interaction of several people over a period of time to achieve a set of common goals. Whitehead et al. (2010) affirm that "collaboration is pervasive throughout software engineering", because almost all non-trivial software projects require effort and talent of many people working together. The team must communicate, share knowledge and artifacts, and coordinate their work. People play a major role in the success of software projects because they participate in the entire lifecycle acting in different roles (customers, users, managers, analysts, developers, etc.) (Altmann and Pomberger, 1999; DeMarco and Lister, 1999; Mistrik et al., 2010; Yildrim, 2006).

Regardless of all known benefits, achieving effective team collaboration remains a challenge. Despite recognizing that collaboration is advantageous, many organizations still do not know how to properly encourage it (Borrelli et al., 1995). Software engineering (SE) researchers still discuss which practices, processes, and tools are able to foster collaboration and to monitor its implementation (Araujo and Borges, 2007; Mistrik et al., 2010). Like other process management domains (Fischer, 2011; Schönthaler et al., 2012), collaboration is key for organizations to cope with the challenges of the modern business environment, which includes flexibility, adaptation, and open innovation (Chesbrough, 2003).

This fact is reinforced by research on distributed software development (DSD) (Cataldo and Herbsleb, 2008; Herbsleb et al., 2005), agile methods (Beck et al., 2001), free/open source software (FOSS) (Raymond, 2001), and ecosystems (Bosch and Bosch-Sijtsema, 2010). In these research areas, the focus on people (their talents, skills and knowledge) and concern with collaboration and communication appear recurrently.

On the other hand, collaboration should only be adopted when it has the potential to produce better results than individuals working alone (Hansen, 2009). In some cases, substantial time and resources are devoted to increase collaboration among team members, but end up being consumed without yielding the desired benefits. Therefore, it is important to determine when collaboration is truly needed and in what intensity (Hansen, 2009). We argue that collaboration can be systematically encouraged in software development organizations by explicitly considering it as part of organization processes.

A **software process** based approach can be useful to foster collaboration. The assumption that the adopted software process directly influences the quality of the developed product has motivated many organizations to invest in defining the processes which they use to produce software products (Cugola and Ghezzi, 1998; Pressman, 2001; Raman, 2000). A software process is "a coherent set of policies, organizational structures, technologies, procedures, and artifacts required to design, develop, deploy and maintain a software product" (Fuggetta, 2000). In general, a process is a way for the organization to plan the work and resources required to perform it, according to their goals. It defines how the organization works, how their activities should be performed, and the roles that people carry in its execution.

However, process definition initiatives are usually challenged by **diversity** (Lindvall and Rus, 2000; Siebel et al., 2003) in various levels. Different kinds of organizations, projects, development models, people and teams make it harder to define specialized processes to cope with known and new development contexts (Magdaleno et al., 2012). To cope with diversity involves the adaptation to the **context** of projects and teams and **reuse** of past experiences in the definition of software processes. **Process composition** is a way to promote the reuse of knowledge related to software processes (Magdaleno et al., 2012). When composing a process based on smaller and reusable units (such as process components or process lines), one can introduce collaboration as the central aspect of interest to be optimized for the combination of units that will form the process for a given project.

To manage collaboration while composing software processes, we created **COMPOOTIM** (Magdaleno et al., 2012). COMPOOTIM supports collaboration management in software processes. It comprises planning, composing, optimizing, and monitoring of software processes with the goal of maximizing collaboration among the members of a team assigned to develop a software project. Each of these four stages is designed to solve some of the challenges of managing collaboration in software processes and contains some proposed solutions.

This paper is dedicated to the composition and optimization stages and the alternatives investigated to find (close to) optimal combinations of process components that might yield a software process which maximizes collaboration among actors performing its activities. Optimization methods can provide solutions to complex problems and can suggest ways to find acceptable solutions in situations where perfect solutions are theoretically impossible or practically unfeasible (Harman and Jones, 2001). Search-based software engineering (SBSE) consists of a research area that studies software engineering from the perspective of optimization and search problems. It applies heuristic techniques to find approximate (near-optimal) solutions for complex problems whose optimal solution could not be found for all instances, because the search space grows exponentially with the size of those instances (Harman et al., 2009).

The optimization stage in COMPOOTIM includes: i) the formal modeling of collaboration in software processes as a SBSE problem (Harman and Jones, 2001); ii) the definition of a collaboration measurement strategy; and iii) the implementation of heuristic search algorithms to address the problem.

This optimization approach was evaluated in the context of a large oil company in Rio de Janeiro, Brazil. The main goal was to evaluate if COMPOOTIM was able to compose processes optimizing collaboration for three software development projects, according to their contexts. This experiment used random search (RS), hill climbing (HC) and genetic algorithms (GA) to find solutions for the process composition with collaboration optimization (PCCO) problem. Brute Force (BF) was also implemented to be used as a benchmark for comparing the algorithms results for small instances.

Algorithms were compared considering the resulting fitness. Results show evidence that GA is more effective (best fitness results) than HC and RS in the search of solutions for collaboration optimization in software process composition for large instances. However, all algorithms are competitive for small instances and even brute force can be a feasible alternative in such a context. These results were complemented by specialist feedback indicating satisfaction with correctness, diversity of options, adherence to the project context, and size and completeness in accordance with the expectations.

The remainder of this paper is structured as follows. In Section 2, the background in collaboration, process reuse and composition is provided. Section 3 summarizes the COMPOOTIM approach. The process composition with collaboration optimization problem is described in Section 4. Section 5 is dedicated to the design of the experimental study that evaluated the optimization of collaboration in the composition of software processes. In Section 6, the results and threats to validity of the evaluation are analyzed. In Section 7, some related works are discussed. Finally, Section 8 concludes the paper and indicates some opportunities for future work.

2. Background

This section summarizes the main background topics of this work. The first one in Section 2.1 is collaboration and its supporting aspects (communication, coordination, group memory and awareness). Then, in Section 2.2, the software process reuse concept, benefits and approaches are presented. Finally, Section 2.3 is dedicated to software process composition including process components and process lines.

Please cite this article as: A.M. Magdaleno et al., Collaboration optimization in software process composition, The Journal of Systems and Software (2014), http://dx.doi.org/10.1016/j.jss.2014.11.036