Information and Software Technology 67 (2015) 44–64

Contents lists available at ScienceDirect

Information and Software Technology

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

Requirements communication and balancing in large-scale software-intensive product development

J. Pernstål [a,*], T. Gorschek [b], R. Feldt [b], D. Florén [a]

[a] Volvo Car Corporation, SE-405 31 Göteborg, Sweden
[b] Blekinge Institute of Technology, SE-371 79 Karlskrona, Sweden

ARTICLE INFO

Article history:
Received 5 February 2015
Received in revised form 17 June 2015
Accepted 18 June 2015
Available online 25 June 2015

Keywords:
Organizational management and coordination
Requirements engineering
Software engineering
Automotive industry
Case study
Lean and agile product development

ABSTRACT

Context: Several industries developing products on a large-scale are facing major challenges as their products are becoming more and more software-intensive. Whereas software was once considered a detail to be bundled, it has since become an intricate and interdependent part of most products. The advancement of software increases the uncertainty and the interdependencies between development tasks and artifacts. A key success factor is good requirements engineering (RE), and in particular, the challenges of effectively and efficiently coordinating and communicating requirements.
Objective: In this work we present a lightweight RE framework and demonstrate and evaluate its industrial applicability in response to the needs of a Swedish automotive company for improving specific problems in inter-departmental requirements coordination and communication in large-scale development of software-intensive systems.
Method: A case study approach and a dynamic validation were used to develop and evaluate the framework in close collaboration with our industrial partner, involving three real-life cases in an ongoing car project. Experience and feedback were collected through observations when applying the framework and from 10 senior industry professionals in a questionnaire and in-depth follow-up interviews.
Results: The experience and feedback about using the framework revealed that it is relevant and applicable for the industry as well as a useful and efficient way to resolve real problems in coordinating and communicating requirements identified at the case company. However, other concerns, such as accessibility to necessary resources and competences in the early development phases, were identified when using the method, which allowed for earlier pre-emptive action to be taken.
Conclusion: Overall, the experience from using the framework and the positive feedback from industry professionals indicated a feasible framework that is applicable in the industry for improving problems related to coordination and communication of requirements. Based on the promising results, our industrial partner has decided upon further validations of the framework in a large-scale pilot program.

1. Introduction

Software is rapidly becoming a substantial component that is widely seen as the main driver and source of innovations in a number of traditionally hardware-focused industries [1–3]. In the automotive domain, the value of automotive software is expected to grow from 127 billion Euros in 2002 to 316 billion in 2015 [4].

With the relative contribution and size of software, the entire complexity of developing large-scale software-intensive systems escalates, introducing new challenges among development organizations [3,5]. Requirements engineering (RE) is commonly identified as crucial and plays a vital role in the potential success of such development efforts [3,6,7]. In particular, the effectiveness of both formal and informal coordination and communication of requirements across organizational boundaries is critical [8,9]. This was also identified as a core challenge at the Volvo Car Corporation (VCC), a developer of luxury cars based in Sweden, as a part of their continuous improvement programs. The dramatic increase of interacting software-intensive automotive systems had increased the interdependencies among the requirements on functions, systems, and components, leading to more complicated organizational dependencies and needs across a multitude of different engineering disciplines and departments (inter-departmental).

There is a range of different solutions addressing requirements communication and coordination. The importance of reducing interdependencies through modularization [10,11], and adopting

* Corresponding author.
E-mail addresses: jpernsta@volvocars.com (J. Pernstål), tony.gorschek@bth.se (T. Gorschek), robert.feldt@bth.se (R. Feldt), dfloren@volvocars.com (D. Florén).

http://dx.doi.org/10.1016/j.infsof.2015.06.007
0950-5849/

techniques for enhancing requirements specification quality have been emphasized, but the increased complexity has made these solutions less applicable [3,6]. In addition, using standards as a coordination mechanism is a key to manage interdependencies in development work [12,13]. For this, software process improvement (SPI) frameworks, such as CMMI [14], ISO/IEC 15504 (a.k.a SPICE) [15], are central to software-intensive systems development. However, they provide little guidance on what the actual implementation should look like, and adopt a one-sizefits-all view without considering the specific needs of organizations and projects [6,16,17].

Solutions to intensify organizational communication, building on team-based methods [18,19], visual management [20,21], goal-oriented approaches [22], and recent agile methods (e.g., Scrum [23]) have also been suggested. However, to the best of our knowledge, it is unclear whether these approaches can contest with the complexity of the large-scale development of software-intensive systems, taking inter-departmental and multidisciplinary aspects into account [24–26].

In this paper, we present a flexible and lightweight RE framework for improving requirements communication and coordination in the design and development process, called Balancing Requirements and Solution Space (BRASS). BRASS was developed in close collaboration with VCC in order to resolve the company's problems identified in the process of balancing requirements and solutions as reported in [27]. The main problems concerned the insufficient quality of requirements and solutions specifications and the lack of communication in the early phases of the development cycle. The importance of solving these problems at VCC is high as they are causing costly loop backs in the final part of the project and inefficient solutions. The cost for this over the life cycle of a car model, which is typically seven years [3], could be estimated at roughly 10MUS$ [27]. Using three real cases, we demonstrate how BRASS was applied in practice and evaluate its usefulness as perceived by professionals at VCC.

We use the term balancing in BRASS, as it is established in the RE-process and is well understood in its pragmatic use at VCC (e.g., [28]). The notion of term balancing is broad, including a combination of the following main coordination activities: (1) creating and breaking down different stakeholders' needs (e.g., business objectives, strategies, and requirements) and identifying the necessity of balancing the needs, (2) analyzing and trading-off the stakeholders' needs and specifying understandable and agreed requirements and solutions, and (3) establishing implementation plans and validating how the actual solutions work. The overall goal of the balancing process is to identify and level out the deviations in the stakeholders' needs and deliver specifications and schedule and report validation tasks for evaluating agreed requirements and solutions developed.

Referring to existing literature on RE, the balancing process at VCC covers all parts of the RE-process (elicitation, analysis and negotiation, specification, validation and verification and requirements management) [29,30]. This means that BRASS does not focus on solving problems in a particular part of the RE-process, such as requirements negotiation, but rather takes a comprehensive view from an RE perspective.

The concept of alignment can also be related to the term balancing and has been conceptualized in various ways in different research fields. In information technology (IT) literature, alignment is, for example, defined as the degree to which the mission, objectives, and plans contained in the business strategy are shared and supported by the IT strategy (e.g., [31]). In software development, technologies focusing on the alignment of activities within and between different development phases, for example, between software architecture and implementation (e.g., [32]) and software requirements and testing [33], have been suggested. Even though the terms alignment and balancing have their similarities, we believe that using alignment can be more confusing due to its deviating contextual interpretations and VCCs unawareness of it.

The remainder of the paper is structured as follows. Section 2 presents the research context, the case company, and elaborates on the challenges identified at VCC, and related work. Section 3 describes BRASS and describes how it can be tailored and applied to the cases studied at VCC. The evaluation of BRASS through validation and the results from using it, as well as its limitations, are presented in Section 4. Section 5 presents the conclusions and Section 6 outlines the future work.

2. Background and related work

This section describes VCC and its core challenges and the specific problems identified and addressed in this paper. We also present related work with regard to these challenges.

2.1. Case description

VCC is a premium car manufacturing company and has approximately 22,000 employees all over the world and produces roughly 450,000 cars per year (2011) [34]. New cars are developed in large-scale projects, involving several 100 of person years and billions of dollars. VCC is organized as a matrix organization and uses a traditional plan-based approach characterized by planning everything from the start of the vehicle program and focusing on documentation. For this, VCC uses a stage-gate model [35] with its milestones to govern the development of the complete car and the V-model [36] to present an overview of design and validation and verification of the complete vehicle and inherent functions, systems, and components. The V-model cycle can be iterated several times during the complete vehicle development. Furthermore, the requirements are mainly specified in written text and administrated by computer-aided RE tools (see also Section 3.1.1). Fig. 1 maps the overall phases in the stage-gate model for developing complete vehicles along with the three main coordination activities of the balancing process and the requirements and design levels in the V-model to a generic product development model developed by Peters et al. [37]. This setting is commonly used in the automotive industry [6,20,38]. The main focus of this study is to resolve problems in the balancing process covering the generic phases pre-design development and design and development process.

The breakdown of product requirements follows a top-down process starting with overall business and user requirements (e.g., product strategies, legal demands on reduced gas consumption, and increased safety) derived from the business development process and ending in component requirements on both hardware and software solutions (e.g., engine control software and engine hardware parts like actuators and sensors), via complete vehicle attributes and function/systems requirements. The annual work typically consists of assessing business and customer requirements and refining them to the vehicle attributes and developing functions that accommodate the required attributes. The primary task in the subsequent pre-program study involves the balancing of functions that have been developed with underlying architectural constraints based on the complete vehicle design. During the concept phase, identified problems in the needs of stakeholders (e.g., function and systems owners, and manufacturing engineers) and system solutions are developed, assessed and selected. The prerequisites for designing the systems' software and hardware components are then compiled and finalized in specifications such as software requirement specifications and design prerequisites for hardware. In addition, verification plans and the need of both