The Journal of Systems and Software 111 (2016) 213–227

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

FLOW-assisted value stream mapping in the early phases of large-scale software development

Nauman Bin Ali[a,*], Kai Petersen[a], Kurt Schneider[b]

[a] Blekinge Institute of Technology, Karlskrona, Sweden
[b] Software Engineering Group, Leibniz Universitat Hannover, Hannover, Germany

ARTICLE INFO

Article history:
Received 2 March 2015
Revised 5 August 2015
Accepted 5 October 2015
Available online 22 October 2015

Keywords:
Case study research
Value stream mapping
Lean software development

ABSTRACT

Value stream mapping (VSM) has been successfully applied in the context of software process improvement. However, its current adaptations from *Lean* manufacturing focus mostly on the flow of artifacts and have taken no account of the essential information flows in software development. A solution specifically targeted toward information flow elicitation and modeling is FLOW. This paper aims to propose and evaluate the combination of VSM and FLOW to identify and alleviate information and communication related challenges in large-scale software development. Using case study research, FLOW-assisted VSM was used for a large product at Ericsson AB, Sweden. Both the process and the outcome of FLOW-assisted VSM have been evaluated from the practitioners' perspective. It was noted that FLOW helped to systematically identify challenges and improvements related to information flow. Practitioners responded favorably to the use of VSM and FLOW, acknowledged the realistic nature and impact on the improvement on software quality, and found the overview of the entire process using the FLOW notation very useful. The combination of FLOW and VSM presented in this study was successful in systematically uncovering issues and characterizing their solutions, indicating their practical usefulness for waste removal with a focus on information flow related issues.

1. Introduction

Value stream mapping (VSM) is a *Lean* practice that maps the current state map (CSM), identifies value-adding and non-value adding activities and steps, and helps to create a shared action plan for an improved future state for the process i.e. a future state map (FSM) (Khurum et al., 2014; Rother and Shook, 1999; McManus, 2005; Poppendieck and Poppendieck, 2003). VSM looks at both the material and information flow (Rother and Shook, 1999). In software development, an equivalent analysis of material flow will look at the flow of *"work items"* e.g. a requirement, use case or a user story through the process (referred to as artifact flow). Contrary to simply the scheduling information that is captured in terms of "information flow" for production processes, for software development it will be pertinent to capture documented/verbal, formal and informal communication. Furthermore, information, knowledge and competence required to carry out the value-adding activities in the software development process have to be identified and analyzed.

The current adaptations of VSM have had a high focus on artifact flow, identifying waiting and productive times (Ali et al., 2015; Khurum et al., 2014; McManus, 2005; Mujtaba et al., 2010). No particular challenge occurs here if the communication structure is relatively simple. However, in the more complex setting of the studied organization, it became equally important to focus on the information flow and explicate it since most of the challenges identified with typical VSM analysis were related to information needs and documentation.

The aim is to achieve an information flow that leverages the *Lean* principle of "pull' such that one process produces only what another process needs (Rother and Shook, 1999). The existing guidelines and notation for VSM (Rother and Shook, 1999; McManus, 2005) do not allow capturing the information flow. As a consequence, we cannot identify value-adding and non-value adding activities required to streamline the process of value creation. There is a need for a systematic, lightweight approach that could supplement VSM, without a lot of overhead. Furthermore, the notation used should be simple and intuitive without requiring additional training of practitioners to understand and analyze the information flows visualized using it.

Information flow modeling (FLOW) has been proposed as a systematic method to analyze and improve communication in software development processes by considering all types of experience and information flows in the organization (Stapel et al., 2009).

* Corresponding author. Tel.: +46 455385541.
E-mail addresses: nauman.ali@bth.se, nauman.ali@gmail.com (N.B. Ali), kai.petersen@bth.se (K. Petersen), kurt.schneider@inf.uni-hannover.de (K. Schneider).

http://dx.doi.org/10.1016/j.jss.2015.10.013
0164-1212/

We propose to combine value stream analysis with FLOW. As both share a focus on capturing and improving information flows, their combination can yield potentially useful results. FLOW applies a systematic approach and a structured yet simple graphical notation to identify and capture the flow of information between people, documents, and activities. It covers and combines formal, informal, documented and verbal information flows in a complex communication structure; hence, it can compensate the inability of existing VSM notation to capture this aspect of information flow in software product development.

To evaluate the usefulness and scalability of using FLOW to assist VSM, we conducted case study research in a large-scale product development at Ericsson AB. We conducted six value stream workshops, with two researchers (first two authors) acting as facilitators, and the third author (who is the inventor of FLOW) as an external observer.

The remainder of the paper is structured as follows: Section 2 presents related work. Section 3 describes the research method. Section 4 presents how FLOW can be used to assist in a typical value stream mapping activity. Section 5 reports the application of FLOW-assisted VSM in the case company. Section 6 presents the results of the study and a follow-up six months after its conclusion. Section 7 reflects on the experience of applying FLOW-assisted VSM and Section 8 concludes the paper.

2. Related work

The related work for this study has three main themes: first the broader theme of software process improvement using *Lean* methods, second is the use of VSM and third is the use of FLOW for mapping and improving software development process.

2.1. *Process improvement in* Lean *software development*

The primary focus of lean process improvement is the identification of bottlenecks hindering the efficient flow of the process. Therefore, different measures and visualizations have been proposed.

Staron and Meding (2011) investigated the method and measures employed by Ericsson to identify bottlenecks in the development process. It is stated that the company on the basis of the Lean concepts of VSM and queues developed these methods and measures. With a similar intention as Staron and Wilhelm, Petersen et al. (2014b) have proposed and evaluated visualizations to identify and resolve bottlenecks in the development process and to follow-up on the degree of success of the resolutions. During this study, the time-line analysis (delineating the productive vs. waiting times) and cumulative flow diagrams were used to analyze the process. Cumulative flow diagrams can be used to guide the identification of root-causes to explain the reasons for the observed bottlenecks (cf. Petersen et al., 2014a as an example).

The work by Staron and Meding (2011) acknowledges the underlying contribution of VSM but that was not the object of their study. While the earlier work on identification of bottlenecks by Petersen et al. (2014a) and Petersen et al. (2014b) uses the typical visualizations used in a value stream analysis, but it does not use the VSM as a framework to guide the improvement activity in the organization.

2.2. *Value stream mapping*

Rother and Shook (1999) presented the guidelines to conduct VSM for manufacturing processes where both material and information flow are mapped. Realizing the differences in product development and manufacturing (McManus, 2005) adapted VSM for this new context. McManus (2005) developed a manual for applying VSM for product development, which is the foundation of this work along with its extension by Khurum et al. (2014).

Two applications of VSM were reported in the context of software product development (Khurum et al., 2014; Mujtaba et al., 2010). Mujtaba et al. (2010) have reported a VSM for product customization process where the goal was to reduce the lead-time. Khurum et al. (2014) identified value aspects that are not traditionally considered, but have to be evaluated on a case to case basis, based on what each organization conducting the VSM consider important from their customers' perspective. In both cases, VSM was done on mature products that have been on the market for a number of years. In this study, we apply VSM in the context of a new large-scale software product development distributed across sites in multiple countries.

Poppendieck and Poppendieck (2003) also provide a brief three-step process to do a VSM. However, the overwhelming focus of analysis in these guidelines is on developing a time-line of progress through a process and making a distinction between waiting and productive time (c.f. Khurum et al., 2014; Poppendieck and Poppendieck, 2003; Mujtaba et al., 2010). The aim of *Lean* is to ensure that a process should make only "what" is required and "when" it is required by the next process (Rother and Shook, 1999). The time-line analysis focuses primarily on "when" in the aim above and seems to overlook the "what" aspect i.e. what information needs exist that should be fulfilled.

In this study, we complement the time line based analysis of work items, which is typical in VSM, with the use of FLOW methodology to identify and analyze information flows in the case of large-scale software product development.

2.3. *Information flow modeling*

Many established software process models focus on documents only (Rausch et al., 2005). Verbal or informal communication, and the use of experience in sophisticated activities are, however, often neglected. In industrial reality, however, many requirements, decisions, and rationales are communicated informally, via phone, meetings, or personal email. The FLOW method offers a graphical notation (Fig. 3) and a modeling approach that covers and combines documented (solid) and non-documented, verbal, or informal (fluid) flow of information. Traditional plan-driven software development tends toward solid information propagation, whereas agile or flexible approaches rely on direct communication (Cockburn, 2002), of fluid information.

Both solid and fluid information flows have complementary advantages and weaknesses. Solid information is typically stored in documents, files, recordings, or other kinds of permanent media. Creating and retrieving information takes longer and requires more effort, but does not rely on the creator, and can be performed at a later point in time and by many people. Fluid information flows faster and often more easily: phone calls, chats, or short messages do not get formally documented, but convey information as well. Fluid information is typically stored in a person's mind. Therefore, a face symbol is used to visualize it.

Many companies need to mix and adjust elements from both ends of the spectrum and try to find an overall optimum. FLOW was created to capture, visualize, and improve situations consisting of a complex network of fluid and solid information flows. Experience is considered an important special case of knowledge that must be present when difficult activities occur and important information is routed through the organization (Stapel and Schneider, 2014).

At the beginning of a FLOW analysis, the current communication channels and network of information flows must be identified. The FLOW method (Stapel and Schneider, 2012; 2014) offers a technique based on interviews. This technique has been applied to medium-sized groups in several companies. Rather complex networks of solid and fluid information and experience flows were identified, modeled, and discussed with domain experts. In one case, a repository was mined as an indicator of document-based (solid)