The Journal of Systems and Software 87 (2014) 48–59

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

Modeling continuous integration practice differences in industry software development

Daniel Ståhl[a,*], Jan Bosch[b]

[a] Ericsson AB, Datalinjen 3, 581 12 Linköping, Sweden
[b] Chalmers Institute of Technology, Göteborg, Sweden

ARTICLE INFO

Article history:
Received 1 February 2013
Received in revised form 23 August 2013
Accepted 29 August 2013
Available online 8 September 2013

Keywords:
Continuous integration
Agile software development

ABSTRACT

Continuous integration is a software practice where developers integrate frequently, at least daily. While this is an ostensibly simple concept, it does leave ample room for interpretation: what is it the developers integrate with, what happens when they do, and what happens before they do? These are all open questions with regards to the details of how one implements the practice of continuous integration, and it is conceivable that not all such implementations in the industry are alike. In this paper we show through a literature review that there are differences in how the practice of continuous integration is interpreted and implemented from case to case. Based on these findings we propose a descriptive model for documenting and thereby better understanding implementations of the continuous integration practice and their differences. The application of the model to an industry software development project is then described in an illustrative case study.

1. Introduction

Continuous integration has, not least as one of the extreme programming practices (Beck, 1999), become popular in software development. It is reported to improve release frequency and predictability (Goodman and Elbaz, 2008), increase developer productivity (Miller, 2008) and improve communication (Downs et al., 2010), among other benefits. In previous work we found that the proposed benefits of continuous integration are disparate not only in literature: there are also great differences in the extent to which practitioners in industry software development projects have experienced those benefits (Ståhl and Bosch, 2013). Consequently, we asked ourselves whether this disparity might be because of differences in the way the continuous integration practice itself had been implemented in different projects, be it because the concept had been interpreted differently or because the project context restricted the freedom of that implementation. Indeed, among the four projects included in the study there were indications that this may be the case, but as that study was not intended for this new research question it did not contain sufficient data to satisfactorily answer whether such differences manifest in software development at large.

Consequently, we decided to establish whether there are also differences in continuous integration descriptions found in the literature, and if so, in which regards the described implementations differ. In this paper we show the results of the systematic review conducted in order to answer this question, along with a proposed descriptive model for continuous integration practice variants based on its findings.

In this work we have focused on process related differences, rather than differences in tooling. While we recognize that tooling may improve or otherwise affect a continuous integration implementation, the practice of continuous integration itself requires no particular tools at all (Fowler, 2013). Consequently we regard tools to be of secondary importance, but not of primary interest. Furthermore, we have not included contextual factors such as the size and longevity of the projects, the business environment or similar parameters. While they may conceivably correlate with variations in continuous integration practice – indeed, we consider the investigation of such correlations an important field of study in itself – they are not themselves aspects of continuous integration.

The contribution of this paper is twofold. First, it shows that there is not one homogeneous practice of continuous integration. Rather there are variation points – those evident in literature are presented and discussed individually in this article – with the term continuous integration acting as an umbrella for a number of variants. This is important, because when consequences of continuous integration are reported and discussed, it must be understood that such consequences potentially may not apply to the practice of continuous integration as a whole, but rather be related to a particular variant of it. Therefore, the second contribution of this article is that a descriptive model that addresses all the variation points uncovered in the study is proposed. Such a model enables better study

* Corresponding author. Tel.: +46 10 711 5573.
E-mail addresses: daniel.stahl@ericsson.com, daniel.b.stahl@gmail.com (D. Ståhl), jan.bosch@chalmers.se (J. Bosch).

0164-1212/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jss.2013.08.032

and evaluation of continuous integration and can thereby bring a finer granularity to our understanding of the practice.

The remainder of this paper is structured as follows. In the next section the research method is described. Then the aspects of continuous integration described in literature, and the statements pertaining to those aspects, are presented and analyzed in Section 3. In Section 4 the proposed model is described, and then applied to a software development project in an illustrative case study in Section 5. The paper is then concluded in Section 6.

2. Research method

The research was conducted by first reviewing existing articles on continuous integration to find differing descriptions of the practice, with the purpose of identifying aspects where there is contention in published literature. In other words, we searched for aspects (represented by clusters of statements, see Section 2.2) where different attributes or characteristics of the practice are evident, as such areas can then be considered to constitute potential variation points. To exemplify, some sources describe how checks and barriers are implemented to prevent non-correctional changes to be integrated on top of a broken build, whereas others relate how anyone is able to contribute anything at any time (see Section 3.2.8). As these are clearly differing views, this area is considered a variation point in the practice of continuous integration.

In contrast, aspects where differing views are either not evident (see e.g. Section 3.1.4) or only addressed by a single source (see e.g. Section 3.1.5) are not regarded as potential variation points, the reasoning being that there appears to be consensus in the industry or that there is insufficient source material to reliably assess them.

Based on this literature review a model for the description and documentation of continuous integration implementations was then constructed, intended as a guide to help ensure that the variation points discovered in the literature review are covered.

2.1. Systematic review

As a result of observations of dramatically different experiences of continuous integration benefits (Ståhl and Bosch, 2013), and our assumption that this may be caused by differences between industry software development projects in how the concept of continuous integration is interpreted and implemented, we wanted to find an answer to the question of "Is there disparity or contention evident in the descriptions of various aspects of the software development practice of continuous integration found in literature?". To answer this question we conducted a systematic review (Kitchenham, 2004), where a review protocol was created and informally reviewed by colleagues. The protocol described the research question above, the sources to be searched (the IEEE Xplore and Inspec databases), the exclusion and inclusion criteria of the review (see Table 1) and the method of extracting and clustering descriptive statements found in the publications (see Section 2.2). Following this the sources were searched (October 2012), with ACM subsequently being added for completeness, for publications relating to the software practice of continuous integration.

The search terms yielded 64, 79 and 45 results in IEEE, Inspec and ACM respectively. Combined, these result sets contained 112 unique items. Exclusion criteria EC1, EC2 and EC3 (see Table 1) were applied to this set, and the abstracts of the remainder were studied to determine whether they dealt with the software practice of continuous integration, or pertained to other fields of research (exclusion criterion EC4). This left a set of 76 publications.

Finally, these 76 publications were reviewed in full in search of descriptions of continuous integration practices (exclusion criterion EC5). Such descriptions were found in 46 of the reviewed articles.

Table 1
Inclusion and exclusion criteria of the literature review.

Inclusion criteria	
IC1	Papers, technical reports, theses, industry white papers and presentations with the terms "continuous integration" and "software" in their titles or abstracts
Exclusion criteria	
EC1	Where studies were published multiple times (e.g. first as a conference paper and then as a journal article) only the most recent publication was included
EC2	Material not available to us in English or Swedish
EC3	Posters for industry talks lacking content beyond abstract and/or references
EC4	Material that does not address the software practice of continuous integration, or only mentions it in passing
EC5	Material that does not describe one or more aspects of how continuous integration practices are, can or should be implemented

2.2. Analysis of literature

Statements as to the nature of continuous integration found in the 46 publications of the literature review were extracted and clustered in groups addressing similar aspects, where one statement may be included in more than one cluster. This yielded 180 discrete, descriptive statements pertaining to one or more aspect of continuous integration and 22 clusters (see Table 2). Following this, any group not containing any disparity in their statements were culled. In other words, only groups of statements describing aspects of continuous integration where contention was evident were preserved. This could either manifest as multiple statements in disagreement, or as statements themselves identifying disparity. Additionally, clusters containing statements from only one unique source were culled.

It shall be noted that determining what in this context constitutes an aspect of continuous integration practice – and thereby a cluster – is ultimately a call of judgment. Particularly, automation is not included, even though it is frequently brought up by papers discussing continuous integration, e.g. stating that "test cases [...] will be folded into the automated regression test suite" (Sturdevant, 2007), that "an automated integration server not only makes integration easy, it also guarantees that an integration build will happen" (Rogers, 2004), "the build process has to be fully automated" (Dösinger et al., 2012) or that "the build process is initiated automatically" (Pesola et al., 2011) to mention a few. For the purposes of this study, we have taken the position that the practice of continuous integration is by definition automated, as described my Martin Fowler: "Each integration is verified by an automated build" (Fowler, 2013). Indeed, from the literature included in this study, we have not found reason to reconsider this position. One source goes so far as to consider it a criterion for success that "all [continuous integration] steps pass without error or human intervention" (Rasmusson, 2004), and so questions of e.g. whether test cases are included in automated test suites rather becomes a matter of the scope of continuous integration, which is covered by its own statement cluster (see Section 3.2.13).

2.3. Proposing a model

Based on the analysis of the literature review, a model for documenting continuous integration was created. The purpose of this model was to cover all the statement clusters displaying contention or disparity, thereby answering all the relevant questions that may set one particular instance of continuous integration apart from another, yet at the same time being practical to use.