Information and Software Technology 56 (2014) 623–643

Contents lists available at ScienceDirect

Information and Software Technology

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

Perceived causes of software project failures – An analysis of their relationships

Timo O.A. Lehtinen *, Mika V. Mäntylä, Jari Vanhanen, Juha Itkonen, Casper Lassenius

Department of Computer Science and Engineering, School of Science, Aalto University, P.O. Box 19210, FI-00076 Aalto, Finland

ARTICLE INFO

Article history:
Received 31 May 2013
Received in revised form 13 December 2013
Accepted 17 January 2014
Available online 10 February 2014

Keywords:
Root cause analysis
Cause and effect relationships
Software project failure
Multiple case study

ABSTRACT

Context: Software project failures are common. Even though the reasons for failures have been widely studied, the analysis of their causal relationships is lacking. This creates an illusion that the causes of project failures are unrelated.
Objective: The aim of this study is to conduct in-depth analysis of software project failures in four software product companies in order to understand the causes of failures and their relationships. For each failure, we want to understand which causes, so called bridge causes, interconnect different process areas, and which causes were perceived as the most promising targets for process improvement.
Method: The causes of failures were detected by conducting root cause analysis. For each cause, we classified its type, process area, and interconnectedness to other causes. We quantitatively analyzed which type, process area, and interconnectedness categories (bridge, local) were common among the causes selected as the most feasible targets for process improvement activities. Finally, we qualitatively analyzed the bridge causes in order to find common denominators for the causal relationships interconnecting the process areas.
Results: For each failure, our method identified causal relationships diagrams including 130–185 causes each. All four cases were unique, albeit some similarities occurred. On average, 50% of the causes were bridge causes. Lack of cooperation, weak task backlog, and lack of software testing resources were common bridge causes. Bridge causes, and causes related to tasks, people, and methods were common among the causes perceived as the most feasible targets for process improvement. The causes related to the project environment were frequent, but seldom perceived as feasible targets for process improvement.
Conclusion: Prevention of a software project failure requires a case-specific analysis and controlling causes outside the process area where the failure surfaces. This calls for collaboration between the individuals and managers responsible for different process areas.

1. Introduction

The discipline of software engineering (SE) was born in 1968 due to software project failures [1]. Preventing software project failures is the main objective of software process improvement (SPI) as it aims at lowering the costs of development work, shortening the time to market, and improving product quality [2]. When considering preventive measures, analyzing the causes of failures becomes important as they explain why the failures occur [3]. This requires understanding the causal relationships, i.e., the causes of failures and their effects. Analyzing the causal relationships between the causes helps develop effective and feasible software process improvement ideas [4,5] as it allows extracting cause and effect relationships [6] that can then be used in process improvement.

While trying to understand why a failure occurs, it is important to analyze all relevant areas of work [7]. Prior studies of software project failures [5,8–24] support this, as the causes of failures reported are spread over various areas including project management, requirements engineering, and implementation. Furthermore, prior studies indicate that the causes are interconnected [8,25], i.e., they have causal relationships between one another. Thus, in order to control the causes of failures, it is important to understand their causal relationships.

Even though software project failures and their causes are widely studied [8–18], prior studies have failed to explain and present the causal relationships between the identified causes. In

* Corresponding author. Tel.: +358 40 775 2781.
E-mail addresses: timo.o.lehtinen@aalto.fi (T.O.A. Lehtinen), mika.mantyla@aalto.fi (M.V. Mäntylä), jari.vanhanen@aalto.fi (J. Vanhanen), juha.itkonen@aalto.fi (J. Itkonen), casper.lassenius@aalto.fi (C. Lassenius).

http://dx.doi.org/10.1016/j.infsof.2014.01.015
0950-5849/

the prior studies, the common causes of failures are often presented as lists where the causes are isolated from one another. This unlikely reflects the real case in software companies [8,25]. Surveys [8,12–14,16,26] and interviews [10,11,17] have been commonly used to detect the causes of failures [27], but not to explain the causal relationships between the causes.

In contrast to prior studies, this study utilizes root cause analysis (RCA), allowing explicitly identifying the causal relationships between the identified causes of project failures. RCA is a structured investigation of a problem to detect the causes that need to be controlled [28]. RCA takes the problem as an input and provides a set of perceived causes with causal relationships as an output [4]. It aims to state what the causes of the problem are, where they occur and why they occur. This helps with software process improvement in various contexts [4,5,19,21,23,24,29–36], and across software organizations [31]. There are many RCA methods available. The method that we used to conduct RCA is called ARCA [4].

The definition for a software project failure is problematic, as the term "failure" is perceived to be vague and challenging to measure [27]. McLeod and MacDonell [27] characterize a software project failure as a breakdown in the software project outcome covering a wide variety of definitions. The term "failure" is either directly related to the outcome of the development process or it is multi-dimensional covering technical, economic, behavioral, psychological, political, subjective, contested/negotiated, and temporal interpretations [27]. Ahmad et al. [37] claim that "it may be almost impossible to find agreement about whether a project succeeded or failed". It has happened that the developers perceive the project as a total success and the other stakeholders perceive it as a dramatic failure [38]. Agarwal and Rathod [39] state that a success and a failure are related to the perceptions of project members. They conclude that the perceptions about a success or a failure are often related to fulfilling the project goals [39]. Similar claims are presented by Procaccino et al. [40]. In our terminology, a software project failure means *a recognizable failure to succeed in the cost, schedule, scope, or quality goals of the project*. The "recognizable" refers to a project failure perceived as severe enough to be prevented in the upcoming projects.

In this paper, we present perceived causal relationships between the causes of project failures in four software product companies. Additionally, we discuss which causes the company personnel perceived as the most promising targets for process improvement activities, and how these differ from the other causes. We extend our prior paper on the causes of the failures of software projects [41], which resulted in a general list of the separated causes of project failures. The rest of the paper is structured as follows: Section 2 introduces the theoretical background. We present the common causes of software project failures, the law of causality, and the relationship between RCA and software process improvement. Section 3 presents our research objective, research questions, as well as how the research data was collected and analyzed. Section 4 presents the case study results through the distributions of the causes of failures and their causal relationships that interconnect the process areas. Section 5 answers the research questions and discusses the most interesting findings and threats to their validity. Finally, Section 6 states the conclusions and proposes future work on this topic.

2. Theoretical background

In this section, we first analyze the prior work to point out the common causes of software project failures. Second, we discuss the causality in software engineering. Then, we present a brief review of RCA that we used as a data collection method in our study. Finally, in Section 2.4 we point out the gaps in prior works.

2.1. Common causes of software project failures

In this section, we discuss the common causes of software project failures introduced in prior studies. We consider the following three questions: (1) what causes of software project failures are introduced, (2) where in the development processes do the causes occur, and (3) what is the relationship between the causes? We base our reasoning on a review of software engineering outcome factors introduced by McLeod and MacDonell [27]. The review covers a total of 177 empirical studies published in the years 1996–2006. Additionally, we supplement the review with otherwise missing, but relevant papers on software project failures we found using the Google Scholar and Scopus databases from 1998 to 2012.

Fig. 1 summarizes the common causes of failures presented in the prior studies and Section 2.1.1 elaborates the prior work behind the figure further. The existing software engineering literature on software project failures indicates that the causes of failures are commonly caused by the project environment, tasks, methods, and people. The causes of failures occur in various processes, which include management, sales & requirements, and implementation. Furthermore, the failures are likely an effect of many interconnected causes having causal relationships to one another. However, while considering such relationships, it seems that there is a gap in the prior studies. Thus, it is difficult to conclude how the causes of failures are interconnected.

2.1.1. Causes of failures and affected process areas

McLeod and MacDonell listed factors that affect the outcome of software systems development projects. These include factors related to project environment, people, methods, and tasks. Their findings resulted in a theory indicating that the development and deployment of software systems is a multidimensional process where people and technology are interconnected [27].

The project environment characterizes the environmental conditions and organizational properties [27] that have an impact on the software project outcome. Moløkken-Østvold and Jørgensen [14] indicate that a chaotic environment is a common cause for software project overruns. In the case of software project failure, the project environment is commonly related to the project complexity [18,42,43], organizational factors [37], available assets [12,37,42–44], policies [43], structure [43], business domain [27,37,45] and technology [45].

The people related causes [27] cover social interaction [27,42,45,46], skills [13,42,45,46], and motivation [16,47]. McLeod and MacDonell [27] indicate that social interaction affects the

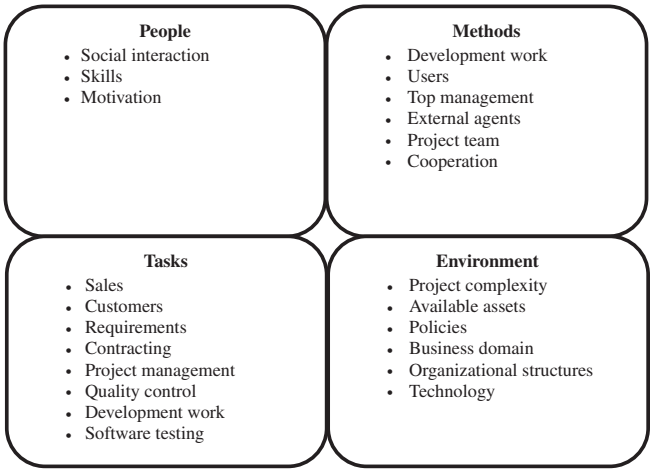

Fig. 1. Summary of the common causes of software project failures.