The Journal of Systems and Software 118 (2016) 85–100

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

A machine learning based software process model recommendation method

Qinbao Song[a,b,*], Xiaoyan Zhu[b], Guangtao Wang[b], Heli Sun[b], He Jiang[c], Chenhao Xue[b], Baowen Xu[a], Wei Song[d]

[a] State Key Laboratory of Software Engineering, Wuhan University, China
[b] Dept. of Computer Science & Technology, Xi'an Jiaotong University, China
[c] School of Software Technology, Dalian University of Technology, China
[d] Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, USA

ARTICLE INFO

Article history:
Received 13 October 2015
Revised 30 April 2016
Accepted 2 May 2016
Available online 4 May 2016

Keywords:
Software project management
Software process model
Model recommendation
Impact analysis
Machine learning

ABSTRACT

Among many factors that influence the success of a software project, the software process model employed is an essential one. An improper process model will be time consuming, error-prone and cost expensive, and further lower the quality of software. Therefore, how to choose an appropriate software process model is a very important problem for software development. Current works focus on the selection criteria and often lead to subjective results. In this paper, we propose a software process model recommendation method, to help project managers choose the most appropriate software process model for a new project at an early stage of development process according to historical software engineering data. The proposed method casts the process model recommendation into a classification problem. It first evaluates the different combinations of the alternative classification and attribute selection algorithms, and the best one is used to build the recommendation model with historical software engineering data; then, the constructed recommendation model is used to predict process models for a new software project with only a few data. We also analyze the mutual impacts between process models and different types of project factors, to further help managers locate the most suitable process model. We found process models are also responsible for defect count, defect severity and software change. Experiments on the data sets from 37 different development teams of different countries show that the average recommendation accuracy of our method reaches up to 82.5%, which makes it potentially useful in practice.

1. Introduction

Software process models provide guidance for systematically coordinating and controlling the tasks that must be performed in order to achieve the project objectives. Thus, among the many factors that influence the success of a software project, the process model employed has been recognized to be one typical characteristic of successful projects (Jones, 1996). An appropriate model helps dealing with the challenges and preventing potential project risks. On the contrary, an unsuitable process model causes additional problems, e.g., time consuming, error-prone, and expensive. Therefore, software process model selection, which is to find good matches between the software process models and actual project situations, for a new project is very important, and emphasized by many researchers (Alexander and Davis, 1991; Stark and Oman, 1997). Over the years, many different kinds of software process models have been proposed, each having different characteristics and areas of suitability, and there is no one process model that fits all the software projects (Kettunen and Laanti, 2005). Therefore, how to choose an appropriate process model objectively among the many different alternatives for a given software project is both very important and potentially useful in practice.

Of the works on software process models, many research investigations compare and contrast the different process models (Abrahamsson et al., 2003; Boehm, 2002; Cockburn, 2000; Sommerville, 1996), and others explore how to choose an appropriate one. The most work on process model selection focuses on the criteria that can be used as attributes to describe the characteristics of process models and projects. Alexander and Davis (1991) presented 20 criteria and a relative rating mechanism for process model selection. Some other criteria were introduced by Egwali and Akwukwuma (2008), Smith (2003), Acuña et al. (2005), and Little (2005). These criteria and selection strategies of process model are heavily based on the experiences of *individual* developers or the characteristics of *some* process models, thus, to

* Corresponding author.
E-mail address: qbsong@mail.xjtu.edu.cn (Q. Song).

http://dx.doi.org/10.1016/j.jss.2016.05.002
0164-1212/

some extent, they are subjective. More importantly, we do not know whether or not the process models selected by these criteria have made software projects successful. Different from the majority work, Kettunen and Laanti (2005) proposed a process model selection frame, which combines knowledge from a variety of literature sources coupled with their practical experiences. Although the frame can only be used as a systematic guide for choosing the process model for large market-driven embedded software products, combining information from multiple resources is a very good strategy.

As we know, in the long history of software development, many software projects were successfully developed. The success of these projects transmits incontrovertible experience on software project management, and which process model is more appropriate for what kind of projects is one of them. These experience came from managers and team members with different expertise, and the projects they worked on could be different in sizes, budget and types, and were developed with varying process models. Thus, these experience cover a wide range of project situations. If fusing these decades of experience together in some way, undoubtedly, this will help managers choosing an appropriate process model for a new project. More importantly, the decision is more objective, because it is determined based on the condensed knowledge of many managers not only the manager oneself.

Thus, in this paper, we make full use of this kind of knowledge and make the following contribution: 1) We propose a software process model recommendation method, to help project managers choose the most appropriate software process model for a new project at an early stage of development process; 2) We analyze the mutual impacts between process models and different types of project factors, to further help managers locate the most suitable process model; 3) We explore if the process models are responsible for defect count, defect severity and software change.

Machine learning can automatically models the relationship between project situations and software process models from historical software project data. So employing machine learning techniques to address the process model selection problem can be an effective solution. In this paper, we present a machine learning based software process model automatic recommendation method to help project managers determine which software process model is more appropriate for a new project at the early stage of development process, according to historical software engineering data.

Our proposed method first automatically chooses project attributes that have the most influence on modeling by attribute selection methods. Since the recommendation is casted as a classification problem, the suitable classification algorithms are then obtained via testing them with sample data sets. After that, the selected classification algorithms are used to construct recommendation model, which finally is employed to choose the appropriate process models.

Since the number of process models chosen by the recommendation model could be more than one, the project managers probably still want to locate the most suitable process model. Therefore, we further analyze the relationship between process models and different types of project factors.

We also study how project- and development team-level factors, which include Project Type, Industry Type, Application Type, Product Character Type, Development Support Type, Project Management Experience, System Experience, Information Technology Experience, Method Experience, Tool Experience and Language Experience, affect process models.

Furthermore, we disclose how process models influence process-level factors such as Change Type, Defect Severity, Defect Count, FTE[1] Staff and Project Effort. Based on the impact analysis, the most suitable process model can be obtained for a new project.

37 different development teams of different countries are used in the experimental study and the results show that: 1) our method is able to recommend suitable process model for a new project; 2)investigation on the mutual impacts between process models and different types of project factors can further help managers locate the most suitable process model; 3) process models are also responsible for defect count, defect severity and software change.

The rest of this article is organized as follows. In Section 2, we discuss the related work. In Section 3.1, we describe the software process model recommendation method. In Section 4, we report the results of an extensive set of experiments. Threats to validity are given in Section 5. Finally, we summarize conclusions in Section 6.

2. Related work

Software process model selection has received substantial attention in recent years. In the past, project managers tended to choose a software process model according to ad hoc and unjustified criteria, which usually lead to an inappropriate process model.

Recently, several methods for process model selection have been proposed. Based on the characteristics of software process models, the related documents of the software projects and the domain knowledge of software development, researchers proposed many criteria to evaluate and differentiate process models. These criteria can further aid project managers to select the process model. Davis et al. (1988) proposed five metrics, and applied them to evaluate the life cycle models and choose the one best satisfying user requirements and having lower life cycle costs. Alexander and Davis (1991) presented 20 criteria and a relative rating mechanism to indicate the appropriateness of the process models to a particular project. Egwali and Akwukwuma (2008) introduced 35 criteria and six function point values for each criterion to choose an appropriate process model that integrated security measures throughout a system's life cycle for a particular project. Smith (2003) proposed 21 project constrains, and applied them on 4 life cycle models. Acuña et al. (2005) also presented various criteria to evaluate the process models, and Little (2005) proposed several attributes to score the suitability of the development process approach for a particular project. Kettunen and Laanti (2005) combined knowledge from a variety of literature sources coupled with their practical experiences to choose the process model for large market-driven embedded software products. These criteria and attributes were found to be helpful in process model selection.

As we mentioned above, most existing approaches for process model selection first adopt some proposed criteria to describe the projects and the process models, then develop some strategy with these criteria to select the suitable process model. Therefore, the criteria and strategy adopted directly affect the selection of process model. However, these criteria were proposed based on the experiences of individual developers or the observations of individual authors, and they are not always consistent. Thus, choosing process models with these criteria lead to subjective or null result.

Moreover, Jiang et al. (2008) adopted the machine learning method, clustering, to help select the suitable requirements engineering techniques according to the clustering rules and received more objective results. Nevertheless, to the best of our knowledge, there has been no work on using the machine learning method to guide the process model selection.

[1] FTE means Full-Time Equivalents.