An Algebraic Approach for Managing Inconsistencies in Software Processes*,**

Qiusong Yang¹,³, Mingshu Li¹,², Qing Wang¹, Guowei Yang¹,³, Jian Zhai¹,³, Juan Li¹, Lishan Hou¹, and Yun Yang⁴

¹ Laboratory for Internet Software Technologies, Institute of Software, Chinese Academy of Sciences, Beijing 100080, China
{qiusong_yang, mingshu, wq, yangguowei, zhaijian, lijuan, houlishan}@itechs.iscas.ac.cn
² State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100080, China
³ Graduate University of Chinese Academy of Sciences, Beijing 100039, China
⁴ Faculty of Information and Communication Technologies, Swinburne University of Technology
Hawthorn, VIC 3122, Melbourne, Australia
yyang@ict.swin.edu.au

Abstract. To produce quality software and evolve them in an economic and timely fashion, enactable software process models are used for regulating development activities with the support of Process-Centered Software Engineering Environments (PCSEEs). However, due to the dynamically changing development environment, the developers do not always follow the process model in presence of unforeseen situations. As human with creativity and variant nature, each developer has his or her own way of doing development that may not be allowed by the process model. As a result, various inconsistencies arise in software processes and then the authority of the process model will be undermined. In this paper, we propose an algebraic approach to promote the efficient management of inconsistencies. With the approach, potential inconsistencies can be precisely detected and valuable diagnostic information is available to help process designers efficiently locate the detected inconsistencies. The effectiveness of the approach is demonstrated by experimenting it on an example process.

Keywords: Software Engineering, Software Process, Inconsistency, Verification, Algebraic, PCSEE, TRISO/ML.

1 Introduction

As for the software process literature, the researches mainly focuses on software process modelling and software process improvement. The research on the software
process modelling involves devising notations for expressing process models, enacting the models within PCSEEs, and providing concrete guidance on the actual development process. To discuss the enactment mechanisms for PCSEEs, Dowson [1] clarifies the three domains of software processes: process definition (or process model in this paper) contains characterizations of processes or fragments of processes expressed in some notation; process performance encompasses the actual activities or actions conducted by human agents and non-human agents in the course of a software project; process definition enactment (or process enactment for short) encompasses what takes place in a PCSEE to support process performance governed by process definition.

In an ideal world, the process enactment can obtain timely and correct feedback from the process performance to know what actual activities or actions are conducted. A software process model describes an ideal process for development and provides procedures to handle possible exceptions. However, the feedback from process performance to process enactment is subject to the variant nature of human and tends to be delayed, ignored, or even erroneous [1]. In addition, it is impossible to define an ideal software process in advance and specify procedures to manage all unforeseen situations. As a result, the environment-level inconsistency [2] will occur when the process performance is not properly reflected in the process enactment.

When a software process is modelled, a set of properties or invariants can be specified to characterize the correctness of process models. When a property is violated in process performance, an inconsistency will arise. This type of internal inconsistency in process performance is called domain-level inconsistency in [2]. A domain-level inconsistency does not necessarily result in an environment-level inconsistency. If the process model successfully predicts the domain-level inconsistency in the process performance, the process enactment will take corresponding actions and the process performance is still faithfully reflected in the process enactment.

As shown in Figure 1, the ultimate goal of PCSEE is to make the process performance governed by the process model. PCSEEs provide mechanisms to enact the process model and components to interact with the environment so that the process model is enforced in the process performance. However, the process performance may deviate from the process model as a result of the existence of inconsistencies. The “performance model” in Figure 1 denotes the underlying model that governs the process performance,

![Fig. 1. Process Inconsistencies](image-url)