Defect Detection Using Event-Based Process Analysis in (Software+) Engineering Projects

Christian Doppler Laboratory for Software Engineering Integration for Flexible Automation Systems
Vienna University of Technology, Austria
{wikan, biffl, tmoser, schatten, weippl, winkler} @ifs.tuwien.ac.at

1. Introduction

Modern software-based systems, like industrial automation systems typically involve the cooperation of several engineering fields, e.g., mechanical, electrical, and software engineering [1]. We call this kind of cooperation “(software+) engineering projects”, as software engineering increasingly provides added value to the resulting software-intensive systems and also depends on the seamless collaboration with all other systems engineering disciplines. In (software+) engineering projects a wide range of heterogeneous tools and data models are used by the engineers, which make early defect detection challenging due to semantic and technical gaps between the selected tools and data models. An integration platform, like the engineering service bus [1], can provide the foundations to overcome these gaps but needs to be extended with a process-oriented view for effective defect detection.

To assure and improve the quality of the engineered system, project and quality managers need a comprehensive view on verification and validation of the system. Currently, individual and selective quality assurance (QA) methods are applied to the system with only limited scope. This approach is time-consuming and requires acceptance tests, which identify defects late in the engineering process. If the system is changed, the QA applied to detect defects as results of these changes is often insufficient and not systematic.

Major challenges in (software+) engineering quality management include engineering model version and change management, early defect detection across engineering discipline and tool boundaries, and engineering process analysis to identify the sources of defects.

In this paper, we address one of these challenges, identifying sources of defects in (software+) engineering projects, by analyzing event data coming from (software+) engineering projects. For an initial evaluation, we study defects in the use case “continuous integration and test” (CI&T), a standard software engineering (SE) process that is implemented rigidly in modern SE environments.

2. Related Work

Process analysis approach has been applied to engineering systems, like workflow management systems, Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) [2]. Schonenberg et al. continue the research on performance analysis based on process mining by proposing a recommendation service to the Process Aware Information Systems (PAISs) [3]. This approach is based on similar past process executions by considering the specific optimization goals. Risk analysis to experimental results of the system can also be applied to detecting defects in (software +) engineering processes.

3. Research Issues

Technical and semantic integration provide the foundation for integrated QA of engineering processes in (software) engineering [1] to detect defects earlier in the engineering process. The next task is to investigate a process analysis approach to detect defects across different engineering disciplines and across tool boundaries that builds on this foundation.

We will discuss the requirements for (software+) engineering process analysis and how it can facilitate the detection of defects and their sources. The tasks include defining the types of defects to be collected, the collection of suitable process data (e.g., engineering tool events), the aggregation and transformation of the collected process data and finally providing the result of the defect detection based on the process analysis approach.
4. Solution Approach

Process Analysis Requirements. In order to analyze engineering processes of (software+) engineering systems, we need to define first the use case to which we apply the experiment and perform the analysis. This is required since certain detail tasks to analyze could differ from one use case to other use cases. Use case definition may simplify the setting of the goals of the analysis, e.g., what kind of defects should be addressed (e.g., detected, localized, or recovered). Process data from the system is also required for analysis. These data can be collected from event logs or communication data between different engineering fields of the system (e.g., data from chat server, mail server, etc.). Some integration and transformation needs to be applied to the data in order to get clean and well-formed data for further analysis. Finally, methods and tools to analyze data need to be chosen and applied.

CI&T Use Case. Figure 1 illustrates an overview of the proposed defect prediction approach using event-based process analysis. Event data originating from heterogeneous engineering tools is collected and integrated in the event log (3) by using an Engineering Knowledge Base [4] (2) and the Engineering Service Bus [1] (1). By using a process analysis tool [5] (4), we can provide information for detecting potential sources of defects (5) in (software+) engineering project environments.

The CI&T use case is a standard life cycle process for SE consisting of several steps: (a) On change of a code unit, build the source code by using a build server, (b) Test the built source code by using a test tool, (c) Package and deploy the compiled source code by using a source code management system. The build result will be published on a project website and if there are errors, a notification will be sent to a list of recipients. This process is implemented by integration build served like Hudson or Continuum in a rigid way.

5. Conclusion

Current status of the work is the building of research prototypes of relevant engineering process support systems and analyzing the results of systematic test runs. Early results with the CI&T process indicate that process analysis is helpful for testing distributed engineering systems that use a common infrastructure for collecting relevant process events. In addition to tracing the expected process flows in the analysis, we also found symptoms of unexpected behavior that helped find defects in the engineering system.

References