Week 8: Reproducing Quality Feedback Loop

Nate Bachmeier

TIM-7140:Software Engineering

April 18, 2021

Northcentral University

# Reproducing Quality Feedback Loop

Businesses of all shapes and sizes are leveraging software-based solutions to reduce costs and become more competitive. This vast breadth of use-cases comes with unique constraints that span technical, cultural, and regulatory requirements. For instance, updating an eCommerce website multiple times per day is trivial compared to NASA's challenges with their Mars Rover. Organizations manage these differences by aligning their software engineering models with the business needs. Regardless of the methodology, all companies desire quality software that gives customers a positive experience. Delivering this outcome is highly challenging and requires multi-discipline skills that span quality assurance and project management.

## Problem Description

SoftTeam is a software consulting firm with fourteen hundred employees that operate across several European nations. They need a mechanism that promotes agile methodologies, ensures quality, and provides task-level visibility. The business began with a standard scrum implementation with sprints, user stories, and defect backlogs. However, the existing tooling did not provide a single-pane view into the system. Instead, engineers would sprawl project status information across source control (e.g., Git and SVN), task tracking (e.g., Jira), and continuous deployment & integration services (CICD, e.g., Jenkins).

## Study Accomplishments

The Developer Operations Team (DevOps) built Q-Rapid, a dashboarding solution for holistically gaining insights into these systems. Business leaders can monitor the Quality Feedback Loop (QFL) across each engineering stage. For instance, when an engineer commits a revision into source control, it triggers a refresh of code quality metrics (e.g., comment ratios and security warnings). Automation and standard open-source tooling produce most of these metrics, ensuring that information is timely and accurate. Teams can also reference quality requirement documentation such as Mantis defects, Jira user stories, and Open Project timelines.

## Methodology

SoftTeam supports Model-IO, a customer-facing application that converts various software diagrams (e.g., Unified Modeling Language) into Java code. After four years of development, the program contains numerous components with varying quality. The DevOps team chose this complex product as the basis of their case study.

## Contributions

## Extensions and Enhancements

# Reproducing the Study

## Setup and Resources

## Methodology

## Limitations and Challenges

## Results

## 