Self-Admitted Technical Debt and Its Practices in Software Builds

Name: Xiao Tao
Laboratory' s name: Software Engineering
Supervisor' s name: Kenichi Matsumoto

Abstract (should be within 1st page)

Technical Debt (TD) is a metaphor used to describe situations where long-term software artifact quality is sacrificed for short-term goals. An explicit representation of such a phenomenon is Self-Admitted Technical Debt (SATD), which refers to debt intentionally introduced and documented as comments by developers. Although significant work has been done on SATD in source code, little is known about SATD in build systems. To address this, this thesis tackles to unveil the implied cost as known as SATD in build systems.

This thesis first characterizes SATD in build systems through a qualitative analysis of 500 SATD comments in the Maven build system of 291 projects. The findings reveal that limitations in tools and libraries and complexities of dependency management are the most frequent causes, accounting for 50% and 24% of the comments, respectively. Developers often document SATD as issues to be fixed later. Additionally, classifiers were trained to detect the two most frequently occurring reasons and four most frequently occurring purposes of SATD, achieving an F1-score of 0.71--0.79. This thesis second investigates the prevalence and characteristics of SATD clones in build systems, examining 50,608 SATD comments from Autotools, CMake, Maven, and Ant build systems. The results show that SATD clones are more prevalent in build systems than in source code, with rates ranging from 62% to 95%. Furthermore, 76% of SATD clone occurrences have high similarity scores, and a quarter of these clones are introduced by the original author. The study identifies external factors as the most frequent locations and limitations in tools and libraries as the most frequent causes of cloned SATD comments.

Together, this thesis also builds a framework for actionable practices of SATD in build systems. This thesis provides a comprehensive understanding of SATD in build systems, highlighting the need for future research on tool support to track and manage SATD backlogs and the design of automated recommendation systems for repaying SATD effectively based on resolved clones.