Abstract

Often we find it difficult to incorporate any changes in a software project during later phases of its development, or during post-delivery maintenance. Primary reason for this is inflexibility in design and code which makes it difficult for changes to be incorporated. This inflexibility substantially increases the cost of making changes and this metaphor has been termed as Technical Debt [1].

While Technical Debt cannot be eliminated completely, its burden needs to be reduced. Many practitioners, especially from agile community, have suggested some practices to avoid or eliminate Technical Debt. This paper discusses on a systematic method to evaluate the six software engineering practices that a developer can follow to minimize Technical Debt. These practices have been used and found to be effective when implemented in projects as discussed here.

References
7. V. Krishna, My Experiments with TDD, ScrumAlliance, 2010; http://www.scrumalliance.org/articles/357-my-experiments-with-tdd

**Index Terms**

Computer Science

Software Engineering

**Keywords**

Technical Credit, Living Budget