CSE 115A - Introduction to Software Engineering

Elijah Hantman

The paper "The Danger of Architectural Technical Debt" By Martini and Bosch, published 2015 at the IEEE/IFIP Conference on Software architecture, is a study into Architectual technical debt and specifically which specific actions are worst.

The paper explains technical debt as decisions made during software development in the interest of meeting business requirements and deadlines. Examples include violating the overall architecture or just making suboptimal choices in general. Techical debt is important because if technical debt is very bad it can slow development putting more pressure on the development team which causes the debt to increase faster and faster. This is a viscious cycle which can kill a project.

The paper covers 5 large international companies looking at 5 Scandinavian sites over the course of 18 months. Overall they list out several ways in which technical debt can be incurred, such as failing to identify non functional requirements early, violating architecture dependencies and creating new dependencies which were not planned or anticipated, using either synchronous or asynchronous functions in the wrong context resulting in poor performance, and so on.

The authors in addition distinguish the kinds of Technical Debt which is most dangerous, accumulating techncial debt. They authors use an example a company which failed to create a standard interface to a database abstraction, this failure resulted in many modules depend on a sub optimal interface which means that changes ripple out to larger and larger portions of the codebase over time.

It is also noted that a common cause of vicious cycles is hidden debt. If the technical debt is noted and consciously kept track of it was much less likely to cause a viscious cycle.

However a major limitation of the paper is that each company had wildly different policies and approaches to Agile development. Different policies can cause technical debt to be worse or better, and can change how technical debt can come about. The best ways to manage technical debt is to first not create it, and second to keep track and carefully manage what debt necessarily is created during the development process.