RAC 3

The paper “The Danger of Architectural Technical Debt: Contagious Debt and Vicious Circles” published in 2015 on 12th Working IEEE/IFIP Conference on Software Architecture by Antonio Martini and Jan Bosch from the Chalmers University of Technology. Showcases the results of a study done about the dangers of Technical Debt, which is the prioritization of short-term goals and obtaining debt which then has to be paid in the long term. Specifically this paper studies the architectural technical debt (ATD), which is when the
code violates the intended architecture for supporting the business goals of the organization in the long term. Also, ADT are dangerous because they can create vicious cycles of debt that can contaminate other aspects of the software development. The authors based their results on an 18-months long exploratory study involving 7 Scandinavian sites in 5 large international software development companies. In addition, the unit of analysis was a subpart of the organization involving at least 10 development teams. Furthermore, the study was divided into several phases that consisted of interviews with employees that had different roles in the organization.

One of the most important points about this study was that it is essential for organizations to monitor ATD items so as to avoid vicious cycles that generate debt for future development as well as ATD items that may contaminate other parts of the development and generate debt for the long term. Another important point is that these items cannot be completely removed as it is inevitable that sometimes to achieve a short term goal a company might resort to creating debt towards their future goals by not doing the best engineering practices. In addition, in the paper it was clear that many of these items that generate debt for future developments are things that can be minimized by implementing good engineering practices, for instance if the code that is used several times is put in its own place that can be accessed by different parts of the development then it is easier to maintain and therefore reduces the possible errors that might come of having repeated code. Furthermore, the paper states that another point that generates ADT’s is when the organization does not implement coding policies and has a uniform way of developing, thus having different variables names and coding pattern that are different which makes it hard for developers to work together. It is also important to mention that the paper clearly states that the phenomenon of ATD has only recently received the attention of the research community and therefore it is hard to understand which ADT to prioritize based on their effects in real situations.

However, the paper makes it clear that ADT are detrimental to the long term success of an organization and that although it is inevitable to eliminate practices that generate these vicious cycles and contagious ADTs, that is is possible to reduce their effects to good engineering practices and constant monitoring these issues.