Learning Better Inspection Optimization Policies

Recent research has shown the value of social metrics for defect prediction. Yet many repositories lack the information required for a social analysis. So, what other means exist to infer how developers interact around their code? One option is static code metrics that have already demonstrated their usefulness in analyzing change in evolving software systems. But do they also help in defect prediction? To address this question we selected a set of static code metrics to determine what classes are most ‘active’ (i.e., the classes where the developers spend much time interacting with each other’s design and implementation decisions) in 33 open-source Java systems that lack details about individual developers. In particular, we assessed the merit of these activity-centric measures in the context of ‘inspection optimization’ (a technique that allows for reading the fewest lines of code in order to find the most defects. For the task of inspection optimization these activity measures perform as well as (usually, within 4%) a theoretical upper bound on the performance of any set of measures. As a result, we argue that activity-centric static code metrics are an excellent predictor for defects.

Keywords: Data Mining; Defect Prediction; Static Measures

Full article:
International Journal of Software Engineering and Knowledge Engineering
Vol. 22, No. 05, pp. 621-644
Markus Lumpe, Rajesh Vasa, Tim Menzies, Burak Turhan