

io-port 06026856

Illes-Seifert, Timea Monika

Justified test foci definition. An empirical approach.

Heidelberg: Univ. Heidelberg, Naturwissenschaftlich-Mathematische Gesamtfakultät (Diss.). 231 p. (2011).

Summary: Since complete testing is not possible, testers have to focus their effort on those parts of the software which they expect to have defects, the test foci. Despite the crucial importance of a systematic and justified definition of the test foci, this task is not well established in practice. Usually, testing resources are uniformly distributed among all parts of the software. A risk of this approach is that parts which contain defects are not sufficiently tested, whereas areas that do not contain defects attain too much consideration. In this thesis, a systematic approach is introduced that allows testers to make justified decisions on the test foci. For this purpose, structural as well as historical characteristics of the software's past releases are analysed visually and statistically in order to find indicators for the software's defects. Structural characteristics refer to the internal structure of the software. This thesis concentrates on the analysis of bad software characteristics, also known as "bad smells". Historical characteristics considered in this thesis are the software's change history and the software's age. Simple and combined analyses of defect variance are introduced in order to determine indicators for defects in software. For this purpose, the defect variance analysis diagram is used to explore the relationship between the software's characteristics and its faultiness visually. Then, statistical procedures are applied in order to determine whether the results obtained visually are statistically significant. The approach is validated in the context of open source development as well as in an industrial setting. For this purpose, seven open source programs as well as several releases of a commercial program are analysed. Thus, the thesis increases the empirical body of knowledge concerning the empirical validation of indicators for defects in software. The results show that there is a subset of bad smells that are well suited as indicators for defects in software. A good indicator in most of all analysed programs is the "God Class" bad smell. Among the historical characteristics analysed in the industrial context, the number of distinct authors as well as the number of changes performed to a file proved to be useful indicators for defects in software.

<http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/2012/13190/>