
Article:
Hierons, R.M. orcid.org/0000-0002-4771-1446 and Xie, T. (2020) On automation in software engineering. Software Testing, Verification and Reliability, 30 (6). e1753. ISSN 0960-0833

https://doi.org/10.1002/stvr.1753

On Automation in Software Engineering

This issue of STVR contains two papers that describe automated techniques. Automation has been an important theme in the software engineering research community for many years. In fact, there is an established annual conference (i.e., ASE) devoted to this topic. Many automated techniques have been developed over the years, and are likely to remain a major focus of work in software testing, verification and reliability. It is clear that some problems (e.g.,, test execution) are more amenable to automation than others (e.g., test generation and correctness-proof construction). However, there has been promising progress in many areas.

The increasing importance and popularity of Artificial Intelligence (AI) has introduced both challenges and opportunities toward automation in software engineering. Systems that use AI often do not have specifications and so it can be difficult to determine whether a behaviour is correct (we have no test oracle) or to prove that a piece of software is correct (prove against what?). In addition, we require a model or specification if we are to apply model-based testing techniques, and classical coverage metrics used in many white-box testing techniques appear not to help. It therefore appears that we will need a completely new set of techniques if we are to extend automation to the complete set of AI systems. The good news is that it may be possible to utilise a range of AI techniques, building on work that uses, for example, metaheuristic search or neural networks to automate software testing and formal verification for (non-AI) software. There is already work in this direction and we look forward to seeing how this area, of using AI to test or verify AI, develops.

In the first paper, Sundeuk Kim, Ilhyun Suh, and Yon Dohn Chung present the simulation-based automatic monitoring (SAM) approach for pinpointing web application failures, including those that require browser APIs or client programs. The SAM approach can monitor all three types of web applications: basic, web browser API-added, and client program-based types. The SAM approach includes a DOM-based simulation model using the installation information of browser API-added and client program-based applications. (Recommended by Sreedevi Sampath).

In the second paper, Thomas Walsh, Gregory M. Kapfhammer, and Phil McMinn address a problem in the area of web page layout. The underlying issue is that the layout of a web pages, as seen by a user, depends upon the device and browser used. Typically, it is not practical to test with all possible combinations and, in addition, the actual layout is usually manually checked. This paper focuses on regression testing and introduces an automated approach that compares the layout of two pages: the page before a change is made and the page after a change is made. Differences are then reported to the developer. (Recommended by Marcio Delamaro).

ROBERT M. HIERONS

Department of Computer Science, The University of Sheffield,

Regent Court, 211 Portobello, Sheffield S1 4DP, UK

TAO XIE

Department of Computer Science and Technology, Peking University,

Beijing, China