Applying Scriptless Test Automation on Web Applications from the Financial Sector

Pekka Aho[1], Govert Buijs[4], Abdurrahman Akın[2], Serafettin Sentürk[5], Fernando Pastor Ricos[3], Stijn de Gouw[1], and Tanja E.J. Vos[1,3]

[1] Open Universiteit, The Netherlands
[2] Research and Development (R&D) Center, Kuveyt Türk Participation Bank Inc., Kocaeli, Turkey
[3] Universitat Politècnica de València, Spain
[4] Erasmus University, The Netherlands
[5] Tübitak Informatics and Information Security Research Center, Kocaeli, Turkey

Abstract. This industry showcase presents experiences on application of TESTAR, an open source tool for scriptless testing through graphical user interface (GUI), to the web applications of Kuveyt Türk Participation Bank in Turkey. Kuveyt Türk Bank uses Selenium and Appium for regression testing of mobile and internet banking, but the maintenance cost of the test scripts is increasing day by day. Therefore, scriptless GUI testing with TESTAR was evaluated. To provide better support for testing web-based applications, TESTAR was extended with Selenium WebDriver integration, JavaScript support, and other new features. Results show that TESTAR detects GUI elements much better after the improvements, and it was able to find 2 relevant errors that were not identified by existing scripted test cases.

Keywords: Automated GUI testing · Industrial case study

1 Problem statement

In traditional scripted test automation, the test cases are defined or generated prior to the test execution. Scripted GUI testing is widely used at industry, for example using Selenium[1] to automate the execution of test cases. However, when the system under testing (SUT) changes during development, the test cases that traverse through the changed part of the SUT must be manually updated, resulting high maintenance costs.

In scriptless test automation, the test cases are dynamically generated during the test execution. Usually scriptless testing involves some level of randomness in the test generation. Whereas in scripted test automation, the test oracles are defined as specific test steps, in scriptless testing the test oracles have to be defined in a more generic way, taking into account that the exact test sequence is not known prior to execution, and all test oracles are checked after each executed action. In this paper, we explore the application of a scriptless GUI testing tool TESTAR to a company of the financial sector by performing a case study following the guidelines defined in [7].

Kuveyt Türk Participation Bank Inc is a private financial institution in Turkey since 1989. It has 414 branch locations across Turkey, and delivers a wide array of financial products and services to customers.

Kuveyt Türk Bank uses Selenium and Appium for regression testing of mobile and internet banking. These tools require manually defined test scripts to run relevant test scenarios in the development environment. The maintenance of scripts must be manually performed for each regression set, provoking high maintenance costs for the bank. The aim of this case study was to evaluate the use of scriptless testing and TESTAR tool in order to reduce the maintenance cost, while increasing test coverage and testing the robustness of systems.

This work has been partially funded by ITEA3 TESTOMAT Project[6], ITEA3 IVVES project[7] and EU H2020 DECODER project[8].

2 Methodology and Technology

TESTAR[9] [8] is an open source tool for scriptless test automation through the graphical user interface (GUI). Originally, TESTAR was developed for testing desktop applications by using the accessibility API of the operating system (OS) for extracting the GUI information [3, 9]. By considering the web browser as a desktop application, it could be used for testing Web applications too [4, 2, 6, 5]. During the industrial study described in this paper, TESTAR has been extended with support for using Selenium WebDriver (WD), instead of accessibility API of the OS, to obtain the state of the GUI.

Before applying TESTAR to the internet banking application, we evaluated the tool on the public website of Kuveyt Türk Bank[10]. The website is more static than the internet banking applications, and acted as an initial SUT to evaluate the logic and capabilities of TESTAR tool. The website is available in multiple languages, but TESTAR was configured to stay on the English part.

When we started with the evaluation, TESTAR was used with the default Windows accessibility API for extracting GUI information, including widgets, their locations and other properties. It was quickly noticed that the accessibility API did not work well with modern Web applications. Without SUT specific instructions, TESTAR did not recognise all the widgets, recognised available actions on widgets that were not visible or clickable, and continued testing after clicking links to external websites. Especially web pages that use DIV-elements or JavaScript to handle user interactions through event listeners were difficult for TESTAR with the standard accessibility API: browsers do not disclose the existence (or absence) of these event listeners. Some of these issues could be solved with SUT-specific instructions in TESTAR protocol class to carry out spe-

[6] www.testomatproject.eu
[7] ivves.eu
[8] www.decoder-project.eu
[9] testar.org
[10] https://www.kuveytturk.com.tr/en/

cific actions, but then these instructions were very specific to the website and would require maintenance if the site was changed.

To address these challenges, TESTAR was extended with various functionalities towards testing of Web pages. The accessibility API was replaced by support for Selenium WebDriver to get information about Web GUIs. To handle widgets that use JavaScript event handlers for user interactions, the Javascript functions to add and remove event handlers were extended dynamically with bookkeeping functionality, using a Monkey Patching solution.

To restrict TESTAR from testing external Web pages, a SUT-specific protocol class was extended with:

1. Allow-listed domain URLs that are to be considered part of the SUT, and
2. Block-listed extensions of resources, such as PDFs, that should not be tested.

Out-of-domain actions and URLs were prohibited, so that TESTAR did not select external links. If it ended up to a domain outside the SUT, a forced action, e.g. browser "go back", was triggered to return back to the SUT. Figure 1 shows the resulting operation flow of TESTAR.

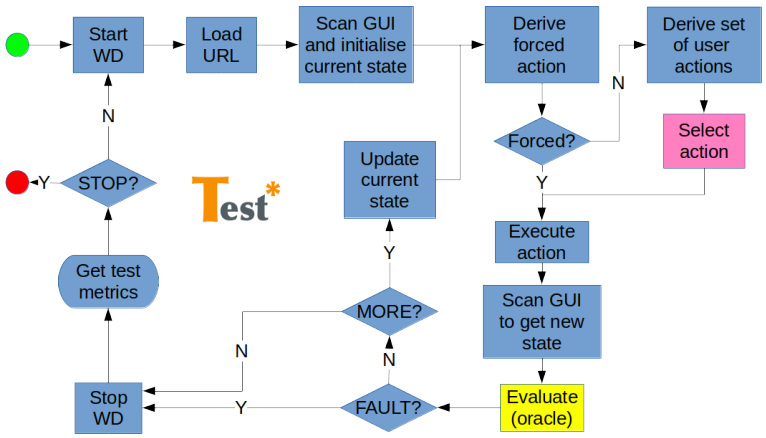

Fig. 1: TESTAR test cycle

3 Results

When evaluating TESTAR with WebDriver, significant improvements were achieved compared to Windows accessibility API. On the public website of Kuveyt Türk Bank, TESTAR with accessibility API found 35 correct widgets that could be interacted with, 155 false positives (widgets that actually could not be interacted with), and missed 14 false negatives (widgets that were not found but could be interacted with). TESTAR with WebDriver and other web extensions found 54 correct widgets, 5 false positives, and 2 false negatives.

The case study continued with testing the more complex internet banking application in the internal development environment of Kuveyt Türk Bank. The initial configuration of TESTAR included passing the login and PIN code process and closing some pop-up warning windows. One detected challenge, relevant for scripted test automation as well, was a unique way to detect specific GUI elements, as element identifiers were dynamically generated and changing, and some other attributes were changing based on the selected language of the web page. Regardless of the challenges, TESTAR was able to find 2 relevant errors that would have been visible for the end users. The errors were fixed by the development team. Scriptless GUI testing complements the existing scripted approaches by covering also the less probable paths through the GUI. However, defining test oracles is more difficult for scriptless approach.

Summarizing, we discovered that Windows accessibility API does not work well with Web applications. We extended TESTAR to use Selenium WebDriver and other Web testing functionalities, and successfully applied it to the industrial case study of the Kuveyt Türk Bank web services, finding 2 relevant errors from the internal development version.

In future, we plan to extend TESTAR further with potentially harmful input generation and adding generic test oracles for web applications, in addition to detecting HTML error codes.

References

1. Selenium homepage. https://www.selenium.dev, last accessed: 21 Apr 2021
2. Almenar, F., Esparcia-Alcázar, A.I., Martínez, M., Rueda, U.: Automated testing of web applications with testar. In: Sarro, F., Deb, K. (eds.) Search Based Software Engineering. pp. 218–223. Springer International Publishing, Cham (2016)
3. Bauersfeld, S., de Rojas, A., Vos, T.E.J.: Evaluating rogue user testing in industry: An experience report. In: Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on. pp. 1–10 (May 2014)
4. Bauersfeld, S., Vos, T.E.J., Condori-Fernández, N., Bagnato, A., Brosse, E.: Evaluating the TESTAR tool in an industrial case study. In: 2014 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '14, Torino, Italy, September 18-19, 2014. p. 4 (2014)
5. Chahim, H., Duran, M., Vos, T.E.J., Aho, P., Condori Fernandez, N.: Scriptless testing at the gui level in an industrial setting. In: Dalpiaz, F., Zdravkovic, J., Loucopoulos, P. (eds.) Proceedings RCIS. pp. 267–284 (2020)
6. Martinez, M., Esparcia, A.I., Rueda, U., Vos, T.E.J., Ortega, C.: Automated localisation testing in industry with testar. In: Wotawa, F., Nica, M., Kushik, N. (eds.) Testing Software and Systems. pp. 241–248. Cham (2016)
7. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical software engineering **14**(2), 131–164 (2009)
8. Vos, T.E.J., Aho, P., Pastor Ricos, F., Rodriguez-Valdes, O., Mulders, A.: Testar – scriptless testing through graphical user interface. Software Testing, Verification and Reliability **31**(3). https://doi.org/10.1002/stvr.1771
9. Vos, T.E., Kruse, P.M., Condori-Fernández, N., Bauersfeld, S., Wegener, J.: Testar: Tool support for test automation at the user interface level **6**(3) (2015)