UNIVERSITY OF TARTU
Institute of Computer Science
Software Engineering Curriculum

Stenver Jerkku

Case study: Improving the Performance of Automated Acceptance Testing with Seleni-um

Master's Thesis (30 ECTS)

Supervisor(s): Dietmar Pfahl
Carlos Paniagua

Tartu 2016

Case study: Improving the Performance of Automated Acceptance Testing with Selenium

Abstract:

In the current world, where everything is on the web, it is nearly a requirement for a company to manage its website. Building a complex website has its costs though – it is difficult and expensive to maintain and develop. Developers often use browser based acceptance tests to assure that the page behaves as dictated by the requirements. The problem with browser based acceptance tests is that they are slow. It is often difficult to use them in continuous integration since the running time of the test suite is so long that the developer moves on to the next task. This increases deployment cycle times, reduces bug-catching rate, makes it hard to trace a bug back to the source commit, and frustrates developers in general.

SaleMove Inc has high standards of quality and follows the TDD principles. In addition to doing TDD, SaleMove has an extensive Selenium-based acceptance test environment, running over 220 tests in 9 different browsers in parallel. Selenium is an automated tool for creating functional tests for web applications [1]. We conducted a case study to analyse and improve SaleMove's acceptance test environment. The goal of the study was to reduce the total acceptance test time of about 50 minutes to around 10 minutes. This allowed us to shorten release cycles, make bug fixing faster and developers happier.

During the study, we analysed the bottlenecks of the SaleMove's acceptance test environment. First, we analysed the test suite and test case startup and the teardown times. Then we identified the bottlenecks of the acceptance test environment and made improvements. We analysed the effect of running Selenium in full cluster mode, i.e., each test running in a clustered hub in parallel with other tests. In this thesis, we describe in detail how we implemented the clustered mode, and how we aggregated test results into a readable format. Finally, we document the gains and costs of this clustered setup and suggest future improvements.

In summary, we managed to shorten SaleMove's acceptance test time to around 13 minutes in full cluster mode. This reduces deployment cycle times, increases developer satisfaction and has other benefits. This comes at the cost of higher complexity of the acceptance test environment and of additional machines that are needed for the clustered mode.

Keywords:

Automated Tests, Selenium, Clustering

CERCS: P170

Juhtumiuuring: Seleniumi automaattestide optimeerimine

Lühikokkuvõte:

Tänapäeval on väga suur osa elust koondunud veebi ning seetõttu on ettevõtted peaaegu et kohustatud oma veebilehtesid korralikult haldama. Samas on veebilehtede arendamine ja ülalpidamine keeruline ja kallis töö. Lehe nõuetekohase käitumise tagamiseks kasutavad arendajad tihtipeale brauseripõhiseid vastuvõtuteste. Brauseripõhiste vastuvõtutestide probleem seisneb aga selles, et need on aeglased. Sageli on nende kasutamine pideva integratsiooni tõttu keeruline, sest vastuvõtutestide käitusaeg on sedavõrd pikk, et arendaja peab enne tagasiside saamist juba järgmise ülesande juurde liikuma. See omakorda pikendab tarneaegu, muudab vigade avastamise ebaefektiivsemaks, raskendab lähtekoodis vigade jälitamist ja vähendab arendajate motivatsiooni.

SaleMove'il on kõrged kvaliteedistandardid ja me järgime TDD põhimõtteid. Lisaks TDD-le on SaleMove'is laiahaardelised vastuvõtutestid – üle 220 unikaalse vastuvõtutesti paralleelselt üheksas erinevas brauseris. Selenium on tööriist, mille abil saab luua automaatseid funktsionaalseid teste veebirakendustele [1]. Juhtumiuuringu eesmärgiks oli analüüsida ja optimeerida SaleMove'i vastuvõtutestide keskkonda. Seadsime sihiks vähendada vastuvõtutestide käitusaega 50 minutilt umbes kümnele minutile. See aitas lühendada tarneaegu, parandada vigu kiiremini ja tõsta arendajate rahulolu.

Uuringu käigus analüüsisime SaleMove'i vastuvõtutestide käitusaja pudelikaelu. Esiteks analüüsisime testjuhtude komplekti ning testi alustamise ja lõpetamise aegu. Seejärel tuvastamise pudelikaelad, optimeerisime vastuvõtutestide keskkonda ning tegime vajalikud parandused. Analüüsisime Seleniumi hajusalt jooksutamist ehk jooksutasime iga testi hajusas keskkonnas paralleelselt teiste testidega. Antud töös kirjeldasime detailselt, kuidas arendasime hajusa keskkonna ja agregeerisime tulemused loetavasse formaati. Viimaseks dokumenteerisime vastuvõtutestide hajusa keskkonna plussid ja miinused ning arutasime, kuidas seda tulevikus veelgi paremaks muuta.

Kokkuvõttes vähendasime SaleMove'i vastuvõtutestide keskkonna käitusaega 50 minutilt 13 minutile ning seda suuresti tänu hajutamisele. See omakorda vähendas meie tarneaegu, suurendas arendajate rahulolu ja oli paljuski muus mõttes kasulik. Samas muutus meie vastuvõtukeskkonna ülesehitus keerukamaks ning lisaks suurenesid hajusa keskkonna tõttu nõuded riistvarale.

Võtmesõnad:

Automaattestid, Selenium, Hajustamine

CERCS: P170

[Text hidden due to license. Contact author for access]

1 Conclusions

In conclusion, setting up massively parallel Selenium Grid acceptance tests is a big, expensive and time-consuming undertaking. If the acceptance tests are not a core part of a company's release cycle, the investment required to do so might not pay off. However, SaleMove is developing a product for Enterprise customers, which creates high demands in quality. Because of this, SaleMove invests a lot in automated testing tools. Furthermore, the initial investment was great, because it helped reduce the release cycle, make developers happier and more involved with acceptance testing, and it helped us catch and fix bugs faster.

After setting up the parallel acceptance test environment, the server costs increased five times, and a lot of man-hours was put into it, but, in the end, the acceptance testing suite run time was decreased by 75%, i.e., from 52 minutes to 13 minutes. The main benefit of reducing the run time can be seen on the deployment days, when everybody is trying to get the tests green. Now that they do not have to wait for almost an hour to see the test results, the bugs can be detected and fixed faster, which in turn helps shorten the release cycles. Eventually, we want to be able to deploy every day. Furthermore, with 52 minutes, the developers rarely, if ever, checked the acceptance tests after merging something in the master. Getting feedback from the tests in just 13 minutes is a lot better, as it keeps the developers engaged, and they can check the results of their changes with only a small delay.

In the future, we plan to optimise the parallel tests, lessening the test run time even further. We are convinced it is possible to get the acceptance tests suite run time to around 10 minutes and maybe even less.

2 Abbreviations

Abbreviation	Meaning
SaaS	Software as a service
TDD	Test driven development
CI	Continuous integration
IP	Internet protocol
CPU	Central processing unit
URI	Uniform resource identifier
IE	Internet Explorer
ORM	Object relationship mapper

3 Works Cited

[1] H. Antawan and K. Marc, “Automating Functional Tests Using Selenium,” in *AGILE 2006 Conference*, Visegrád, 2006.

[2] Z. Zhan, Selenium WebDriver Recipes in C#, Apress, 2015.

[3] G. E. Mills, Action Research: A Guide for the Teacher Researcher., Pearson; 4 edition, 2010.

[4] Yandex, “Grid router,” 12 March 2016. [Online]. Available: https://github.com/seleniumkit/gridrouter.

[5] B. Haugset, “ Automated Acceptance Testing: A Literature Review and an Industrial Case Study,” in *Agile, 2008. AGILE '08. Conference*, Toronto, ON, 2008.

[6] Ruby community, “Ruby-lang,” 1 November 2015. [Online]. Available: https://www.ruby-lang.org. [Accessed 13 November 2015].

[7] Capybara community, “Capybara,” 9 November 2015. [Online]. Available: https://github.com/jnicklas/capybara. [Accessed 9 November 2015].

[8] Rspec community, “Rspec,” 9 November 2015. [Online]. Available: http://rspec.info/. [Accessed 9 November 2015].

[9] Selenium community, “Selenium,” 9 November 2015. [Online]. Available: http://www.seleniumhq.org/. [Accessed 9 November 2015].

[10] Jenkins community, “Jenkins,” 9 November 2015. [Online]. Available: https://jenkins-ci.org/. [Accessed 9 November 2015].

[11] Tikal AML team, “Multijob,” 9 November 2015. [Online]. Available: https://wiki.jenkins-ci.org/display/JENKINS/Multijob+Plugin. [Accessed 9 November 2015].

[12] Amazon Web Services, Inc, “Amazon AWS,” 9 November 2015. [Online]. Available: https://aws.amazon.com/. [Accessed 9 November 2015].

[13] HashiCorp, “Vagrant,” 9 November 2015. [Online]. Available: https://www.vagrantup.com/. [Accessed 9 November 2015].

[14] Oracle Corporation, “Virtualbox,” 9 November 2015. [Online]. Available: https://www.virtualbox.org/. [Accessed 9 November 2015].

[15] Docker, Inc., “Docker,” 9 November 2015. [Online]. Available: https://www.docker.com/. [Accessed 9 November 2015].

[16] Pivotal Software, Inc, “RabbitMq,” 9 November 2015. [Online]. Available: https://www.rabbitmq.com/. [Accessed 9 November 2015].

[17] Rails community, “Ruby on rails,” 9 November 2015. [Online]. Available: http://guides.rubyonrails.org/. [Accessed 9 November 2015].

[18] The PostgreSQL Global Development Group, “PostgreSQL,” 9 November 2015. [Online]. Available: http://www.postgresql.org/. [Accessed 9 November 2015].

[19] MongoDB, Inc., “MongoDB,” 9 November 2015. [Online]. Available: https://www.mongodb.org/. [Accessed 9 November 2015].

[20] Metamarkets Group Inc, “Druid,” 9 November 2015. [Online]. Available: http://druid.io/. [Accessed 9 November 2015].

[21] G. Inc, “WebRTC,” Google Inc, May 8 2016. [Online]. Available: https://webrtc.org/.

[22] I. SaleMove, “ METHOD AND APPARATUS FOR PUSHING APPLICATIONS TO A WEBSITE VISITOR DURING CO-BROWSING”. Patent 8769119, 6 November 2013.

[23] M. Grosser, “Parallel,” 27 March 2016. [Online]. Available: https://github.com/grosser/parallel.

[24] M. Grosser, “Parallel_tests,” 31 March 2016. [Online]. Available: https://github.com/grosser/parallel_tests.

[25] S. Community, “Docker Selenium,” 25 April 2016. [Online]. Available: https://github.com/SeleniumHQ/docker-selenium.

[26] C. D. Community, “Chrome driver,” 14 March 2016. [Online]. Available: https://sites.google.com/a/chromium.org/chromedriver/.

[27] onlywade, “Docker Scaling problems,” 14 August 2016. [Online]. Available: https://github.com/SeleniumHQ/docker-selenium/issues/87.

[28] B. Nolan, “Least Load,” 13 June 2013. [Online]. Available: https://wiki.jenkins-ci.org/display/JENKINS/Least+Load+Plugin.

[29] A. Nikolaenko, “Meet the Selenium Grid,” [Online]. Available: http://www.slideshare.net/alekseytcherezov/meet-the-selenium-grid.

[30] B. Stack, “Browser Stack,” 1 May 2016. [Online]. Available: https://www.browserstack.com.

[31] S. Labs, “Sauce Labs,” 1 May 2016. [Online]. Available: https://saucelabs.com.

[32] S. Gundepuneni, “Services running Selenium on the cloud,” 1 May 2016. [Online]. Available: http://seleniummansion.blogspot.com.ee/2014/10/services-running-selenium-on-cloud.html.

[33] Selenium Community, “Selenium Docker,” 15 February 2016. [Online]. Available: https://github.com/SeleniumHQ/docker-selenium. [Accessed 2 April 2016].

[34] “Research methodology and methods,” thewireframecommunity.com, 25 11 2010. [Online]. Available: http://www.thewireframecommunity.com/node/196. [Accessed 8 May 2016].

Appendix - work done

As part of this thesis we wrote around 2000 lines of Ruby code, 300 lines of bash scripts, 100 lines of Docker build scripts and configured around 40 Jenkins jobs. We also managed 10 Virtual machines in Amazon AWS and around 300 dockers with Selenium browsers or Grid.

I. License

Non-exclusive licence to reproduce thesis

I, **Stenver Jerkku** (date of birth: 10.10.1990),

(*author's name*)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

reproduce, for the purpose of preservation, including for the purpose of preservation in the DSpace digital archives until expiry of the term of validity of the copyright

Case study: Improving the Performance of Automated Acceptance Testing with Selenium ,

(title of thesis)

supervised by Dietmar Pfahl,

(supervisor's name)

2. Making the thesis available to the public is not allowed.

3. I am aware of the fact that the author retains the right refered to in point 1.

4. This is to certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **19.05.2016**