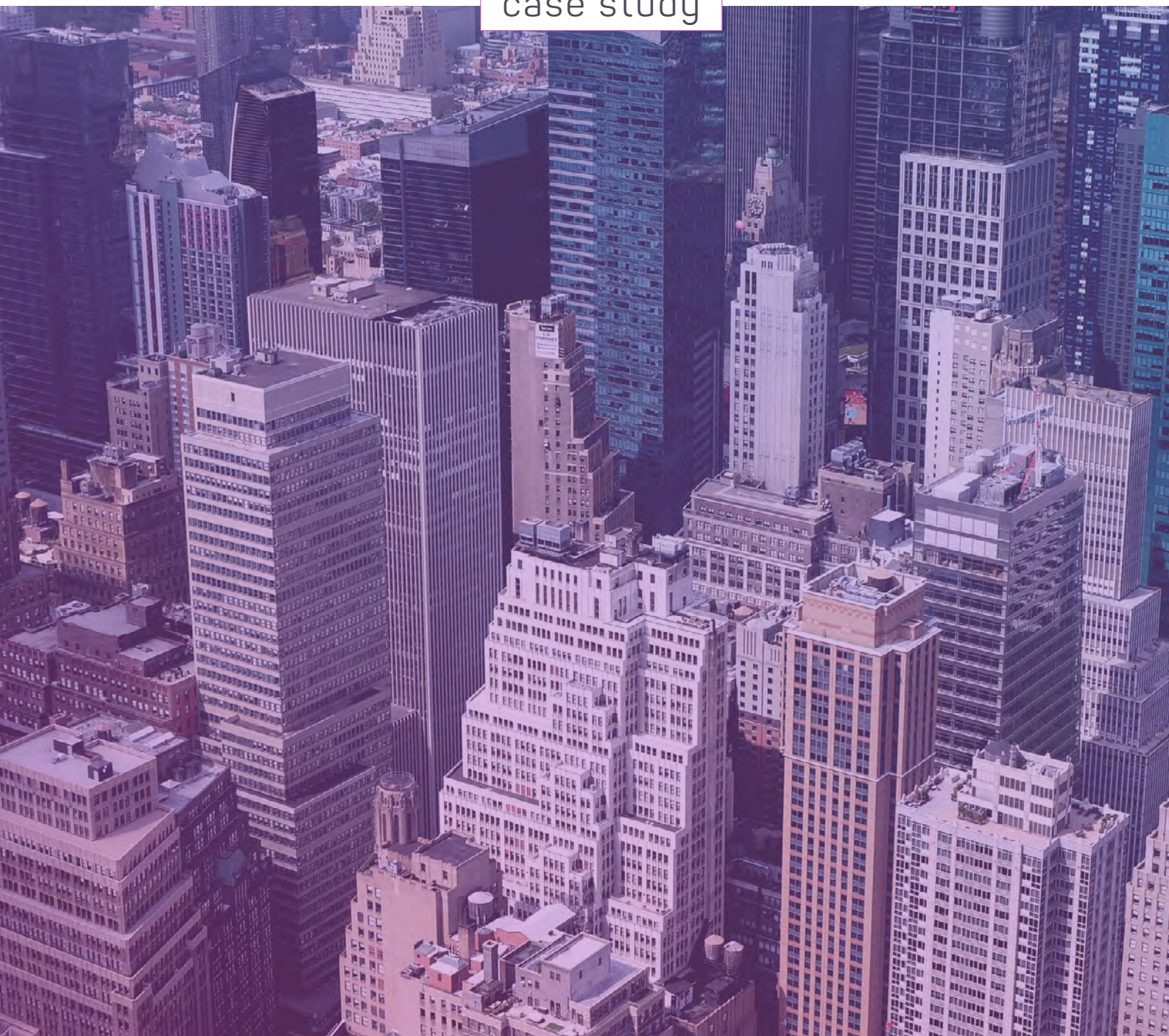
mabl

A FORTUNE 500 SECURITY COMPANY

case study



Describe mabl in 3 words:

"Makes life easy."

This case study features a director of software quality who manages a team of 50 testers, distributed across three different remote locations. His team is evenly split between manual testers and SDETs (software development engineers in test).

While his automation team does a lot of REST, SOAP, and Web Services automation, they traditionally use Python and Java with Selenium on the Amazon Web Services cloud platform for their cross browser testing.

Current Testing Challenges with Selenium

TEST DEVELOPMENT VELOCITY

- Requires experience with Selenium WebDriver
- Requires 2 hours to automate 1 scenario
- Coding experience and UI automation skills are required

MAINTENANCE AND RELIABILITY

- Tests are flaky due to infrastructure issues
- Requires fixes to tests if there is a change in element locators in app UI
- Requires manual adjustment of wait times based on app behaviour

The Challenge

3rd party SaaS apps like Office 365, Google Apps, Salesforce, and Dropbox are commonly used by both enterprises and consumers. One of the projects that he oversees involves a traffic analysis tool which integrates with 3rd party apps. His team must generate traffic from those apps in order to test the output of their traffic analysis tool. One challenge he has run into is not being able to easily automate traffic generation from 3rd party apps quickly and efficiently.

Even with a team of automation experts, he knew it would be difficult to automate tests on 3rd party apps given how brittle Selenium has been for the team, as well as how often the app UIs are updated by the vendors. For example, Google Apps is updated about every 2 weeks, and Office 365 is updated about every 2-3 days. Maintaining the test scripts for these apps would become a nightmare.

On top of the test maintenance burden, his team would need to dedicate more resources to infrastructure administration. On a different project that he oversees that is using Selenium, 1 of the 8 SDETs on the team works full time on maintaining the test infrastructure alone.

> *"We felt like Selenium might not be the best approach since we have limited resources, and we know the pain of Selenium, infrastructure setup, the execution of the tests being expensive, and the flakiness of Selenium tests."*

Because of these challenges, he began searching for an alternative testing solution to Selenium. He found mabl.

The Solution

Because mabl is so easy to use, instead of gathering a team of 8 SDETs, he was able to assign only one junior SDET to this project involving 3rd party apps. They quickly discovered that they were able to create more tests with mabl with significantly less effort than with Selenium.

> *"[The mabl project] has 80 test cases created in a short span of time. If we had to develop the same test cases in Selenium, it would have taken quite a bit of effort. The savings are there in test development."*

mabl solved the challenge of operational overhead. His team working on the Selenium project has a Jenkins cluster with Windows VMs that spin up on schedule. 100 Windows VMs are used on an ongoing basis, and a full time resource is dedicated to administering the testing infrastructure. Windows VMs are slow and unreliable, so issues are constantly cropping up; sometimes the VMs don't spin up, and other times the VM and its configuration is lost altogether. Because mabl completely manages the testing infrastructure, he can assign his SDETs to other projects.

> *"With mabl, we invest 0 on infrastructure."*

mabl has returned more time to the team in its portability of tests. Once tests are created for one environment, they can be easily run on other environments with a simple change in plan configuration. This saves his team from having to recreate and rewrite tests for different browsers.

> *"Selenium itself claims it's browser agnostic, but you do have to do tweaking in the code to make it work on different browsers. With mabl we don't have to make different script changes to run them across mutliple browsers."*

Not only has the automation team benefitted from mabl, but his manual testing team was introduced to mabl. After only one training session, they were able to automate 100 test cases.

Benefits & ROI Summary

Less Test maintenance
- Minimal test maintenance overhead
- Locator changes are handled by mabl

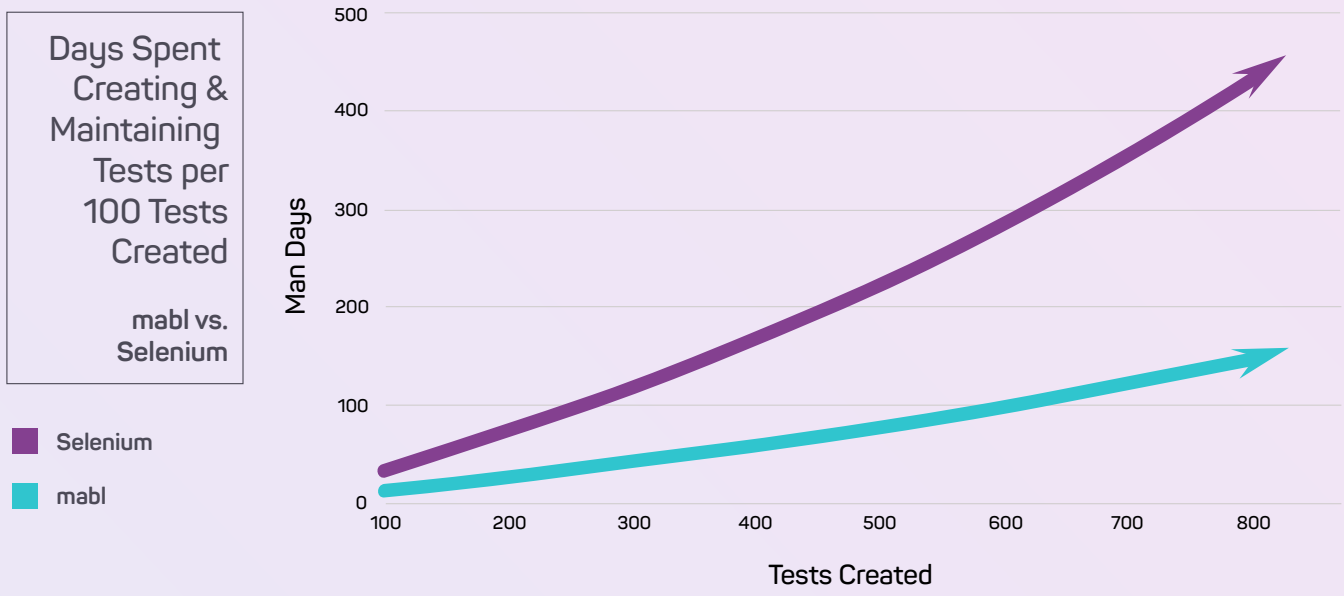
Enables manual testers to automate
- Minimal learning curve for manual testers
- No coding experience needed

Faster Test Development
- Can automate 2 scenarios in 1 hour

No Operational burden
- No Infrastructure is required
- All the test machines and required software is managed by mabl

Days Spent Creating & Maintaining Tests per 100 Tests Created

mabl vs. Selenium



ROI Findings	Time to Implement 100 Tests (hours)	Test Maintenance (hours)	Total Hours Spent	Engineering cost per year*
Selenium on AWS	264	33	297	**$352,000**
mabl	96	10	106	**$44,000**
Savings	**64%**	**70%**	**64%**	**88%**

*Given 8 SDETs on Selenium vs 1 SDET on mabl


mabl
mabl.com
Get a mabl demo