AutomationSTAR

Test Automation Community Survey

Compiled by Dorothy Graham

Abstract

In preparation for my keynote at AutomationSTAR, I wondered what the current state of test automation is, so I asked EuroSTAR to send out a survey for me. What tools are people using? What training have people had in the tools, testing and good automation practices? How easy would it be to change the automation platform or tool? How similar are people's automated tests are to the equivalent manual tests? A total of 206 responses were received between August 24th and September 15th 2024.

I found the responses a fascinating peek into how people are doing automation today. Some things surprised me: 80% of respondents are using open source tools, 82% find out about automation on the job, 25+% had no training at all in either the tool they are using or test automation architecture.

There was one open-ended question, "What is your worst problem with automated testing?" I was expecting maintenance or flaky tests to be the most "popular" problem but the issue mentioned by most respondents was management issues. This included insufficient time, lack of standards, knowing what to automate, but the most popular sub-category was lack of understanding about automation, i.e. unrealistic expectations.

This document goes through the results of the survey in detail, and I hope you will find it as interesting as I did to see where we are in automation in 2024.

Many thanks to Isabel Evans, Chris Loder and Mark Fewster for comments on this survey.

Contents

The Test Automation Community Survey had 9 questions that were multiple-choice, and one that asked for a text answer.

In this document, I will show each question in turn, with the results from the 206 respondents shown as a graph then the numbers followed by my comments.

Three Key Points



Management issues are alive and well, particular unrealistic expectations!



Lessons about how to automate well appear to have been forgotten (or never known about)

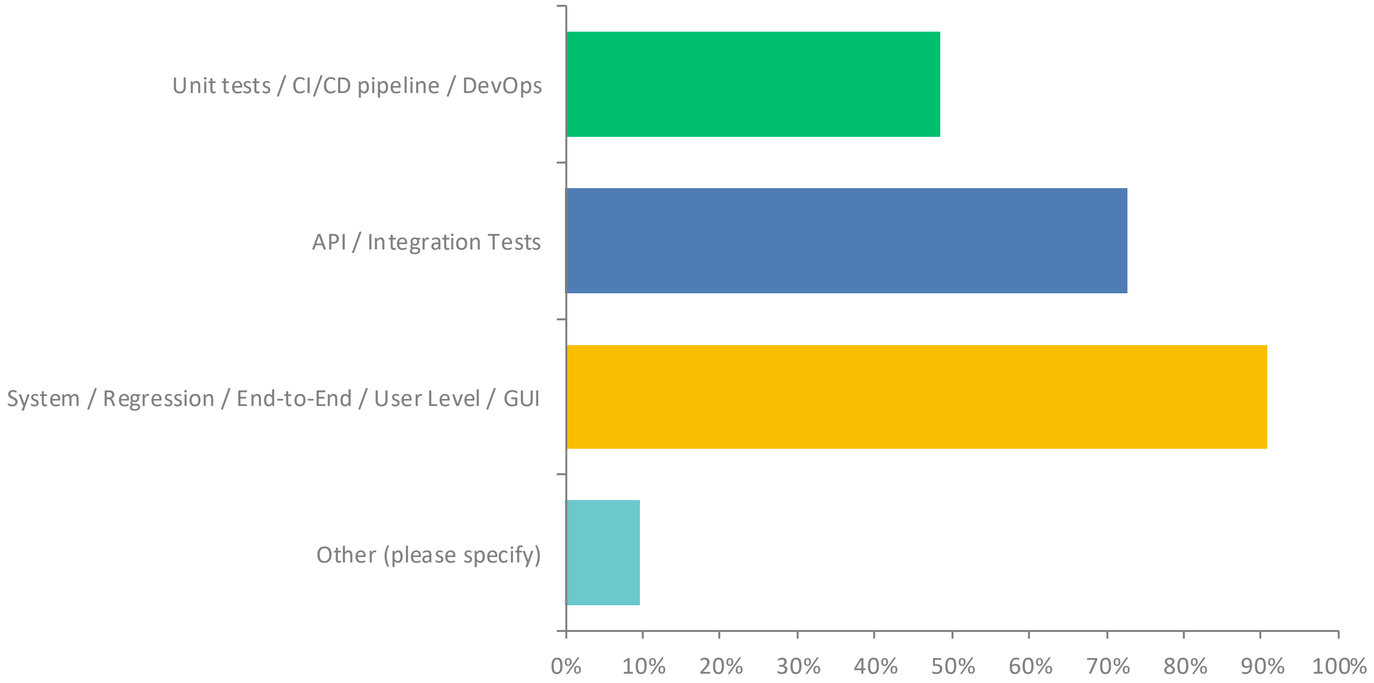


Over 80% of respondents use open source tools

QUESTION 1

What tests are you automating?

(Tick all that apply)



ANSWER CHOICES	RESPONSES	
Unit tests / CI/CD pipeline / DevOps	48.54%	100
API / Integration Tests	72.82%	150
System / Regression / End-to-End / User Level / GUI	90.78%	187
Other (please specify)	9.71%	20
TOTAL		457

Dorothy's Thoughts

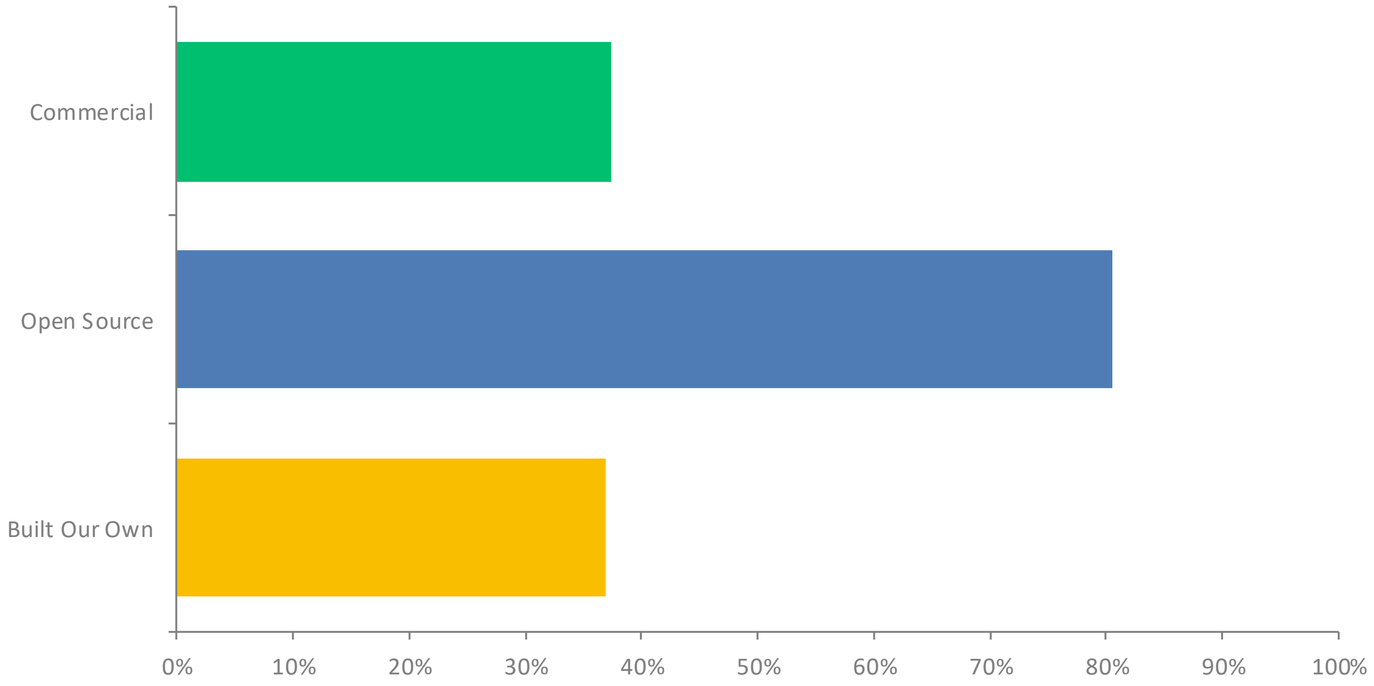
Note that the column “Respondees” has the percentage of the options chosen divided by the total number of respondees (206), not divided by the number of responses for the category (457). So the 187 people who are doing system testing etc are 91% of the 206 respondents.

Although 206 people responded, they could choose more than one option - clearly most people are testing at more than one level of testing. Most people seem to be testing at system level and at least one other level.

The “Other” category included non-functional tests such as performance and security, data tests, infrastructure or contract tests. A few I would have classified as system tests (e.g. E2E tests, smoke tests) and there were some I wasn’t sure of (e.g. combinatorial or manual tests).

QUESTION 2

What test execution automation tool(s) do you use?



ANSWER CHOICES	RESPONSES	
Commercial	37.38%	77
Open Source	80.58%	166
Built Our Own	36.89%	76
TOTAL		319

Dorothy's Thoughts

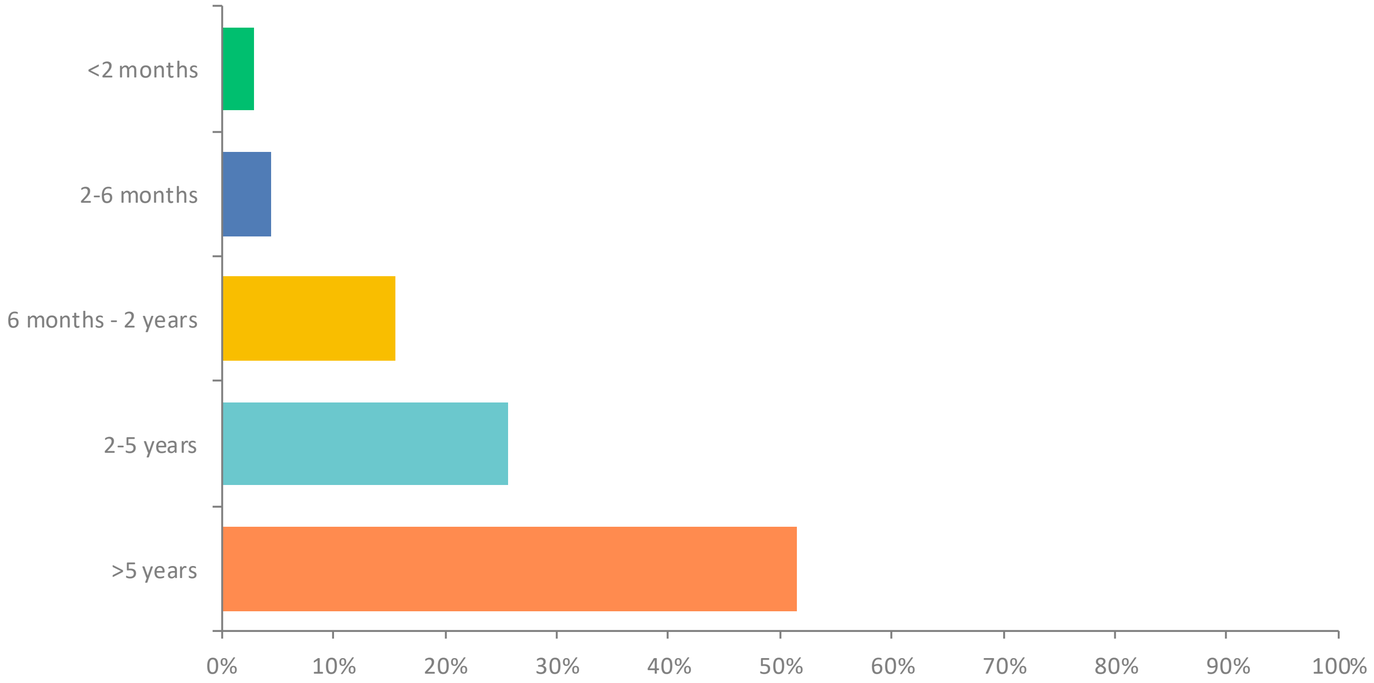
This graph is similar to the previous, with the percentages being calculated for the number of respondents, not the number of responses.

This does show that over 80% are using open source tools, so they are certainly popular! Half of the respondents are using more than one type of tool, and some of those may be using all three types of tool.

QUESTION 3

How long have you been using this/these automation tools?

(Based on the highest level of testing that you do)



ANSWER CHOICES	RESPONSES	
<2 months	2.91%	6
2-6 months	4.37%	9
6 months - 2 years	15.53%	32
2-5 years	25.73%	53
>5 years	51.46%	106
TOTAL		206

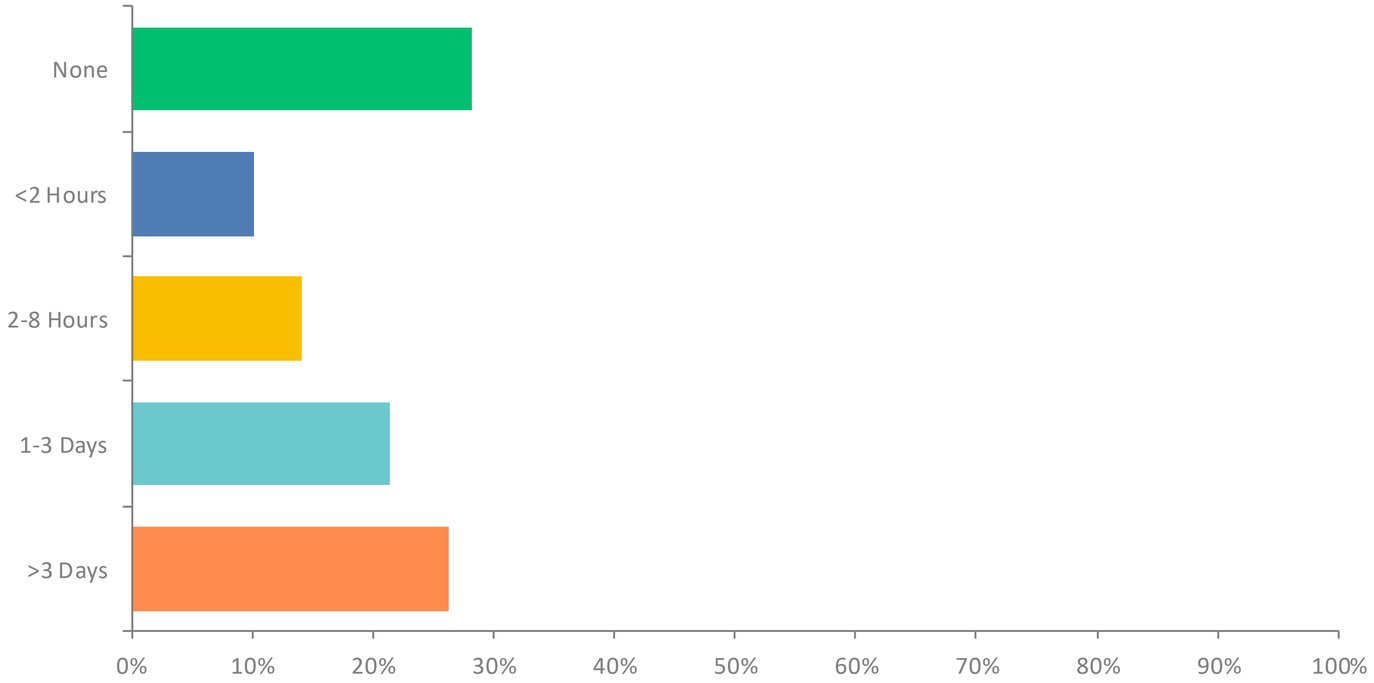
Dorothy's Thoughts

Although there were a few people who are just getting started in automation with a specific tool or tools, more than half have been automating for more than 5 years (possibly using more than one tool). So the problems they are having may not be “teething” problems but perhaps more persistent problems. Isabel Evans tells me that also fits with her findings. (See link at the end of this PDF.)

Of course, just because you have been doing automation for a long time, this doesn’t mean that you are doing it in the best way! Some respondents clearly have been very successful, but others who have been automating for a while have not had the same success.

QUESTION 4

How much training did you have in the use of the tool itself?



ANSWER CHOICES	RESPONSES	
None	28.16%	58
<2 Hours	10.19%	21
2-8 Hours	14.08%	29
1-3 Days	21.36%	44
>3 Days	26.21%	54
TOTAL		206

Dorothy's Thoughts

One reason for including this question was one of the things that Isabel Evans had found in her extensive research (for her PhD) on the use of test tools (see reference at the end of this ebook). She was surprised that people hadn't had training even in the use of the tool itself, and I found that this was true for more than a quarter of the respondents here. It's good that half of the respondents did have at least one day of training (is one day enough?). But more than half had less than a single day's training in a tool that they are presumably using on a daily basis!

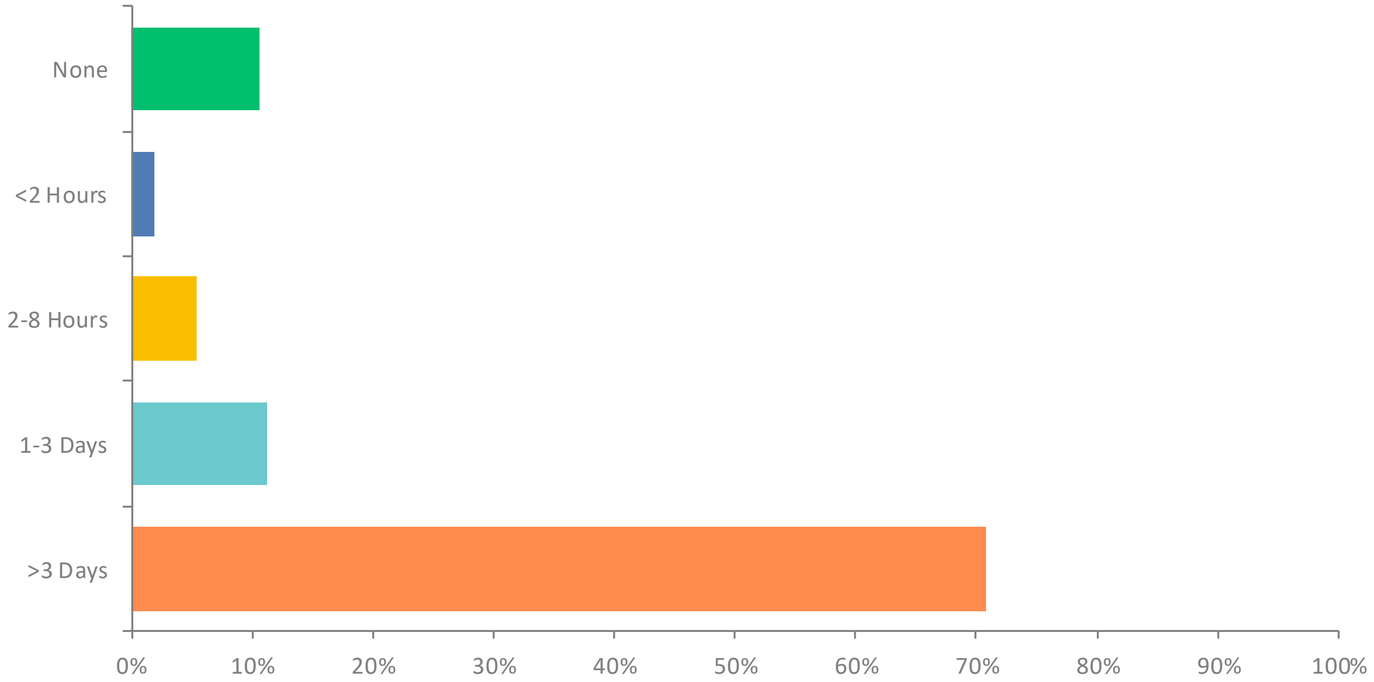
How much training should there be for using something for the first time? When you think back to the last time you did something new for the first time, how good were you at it? (Skiing anyone?) Your first attempts at using a new tool will not be your best efforts - in fact, it is often extremely confusing when you use something new (it's not just me, is it?)

Without even the most basic training, you will make lots of mistakes and take a lot of time trying to find how best to use the tool. Even if you do eventually discover good ways to use the tool, it will be a long and frustrating journey. Good training not only saves time, but also leads you directly to a much better place than you would likely discover yourself. Mark Fewster points out that the best benefits typically come from what only automation can do.

Skimping on training is a false economy.

QUESTION 5

How much training have you had in software testing and test design techniques? (e.g. ISTQB CTFL, RST etc.)



ANSWER CHOICES	RESPONSES	
None	10.68%	22
<2 Hours	1.94%	4
2-8 Hours	5.34%	11
1-3 Days	11.17%	23
>3 Days	70.87%	146
TOTAL		206

Dorothy's Thoughts

This was a bit of a surprise to me - I thought that the respondents would be mainly test automators, who are not necessarily testers, but 71% have had more than 3 days of training in testing itself. Of course, it is useful for automators to know about testing, but I suspect that this shows something else - that most people doing test automation are (or were) testers.

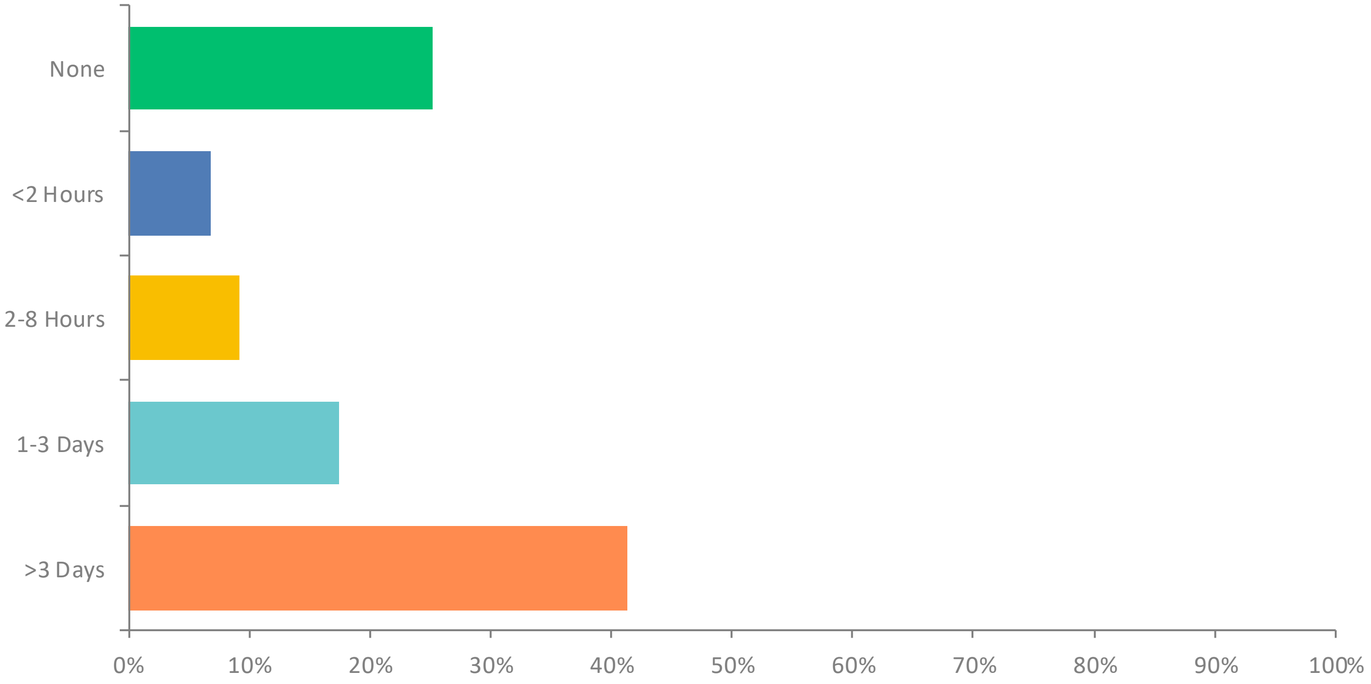
It is a popular view (which I strongly disagree with) that testers should [all] become test automators. I have no problem with people who want to do both testing and automation, but the skills of a tester are different to the skills of an automator. There are some testers who don't want to learn to code, and who may not be very good at it or enjoy it - forcing these testers into an automation role is wrong IMHO. I hope that the testers who are doing automation (the 71% here) are happy to be in an automation role!

It would be interesting to know whether training in testing was useful for automators - but that's for another time.

QUESTION 6

How much training did you have in the design of automated testware or testware architecture?

(e.g. ISTQB Test Automation Engineer or Conference Tutorial)



ANSWER CHOICES	RESPONSES	
None	25.24%	52
<2 Hours	6.80%	14
2-8 Hours	9.22%	19
1-3 Days	17.48%	36
>3 Days	41.26%	85
TOTAL		206

Dorothy's Thoughts

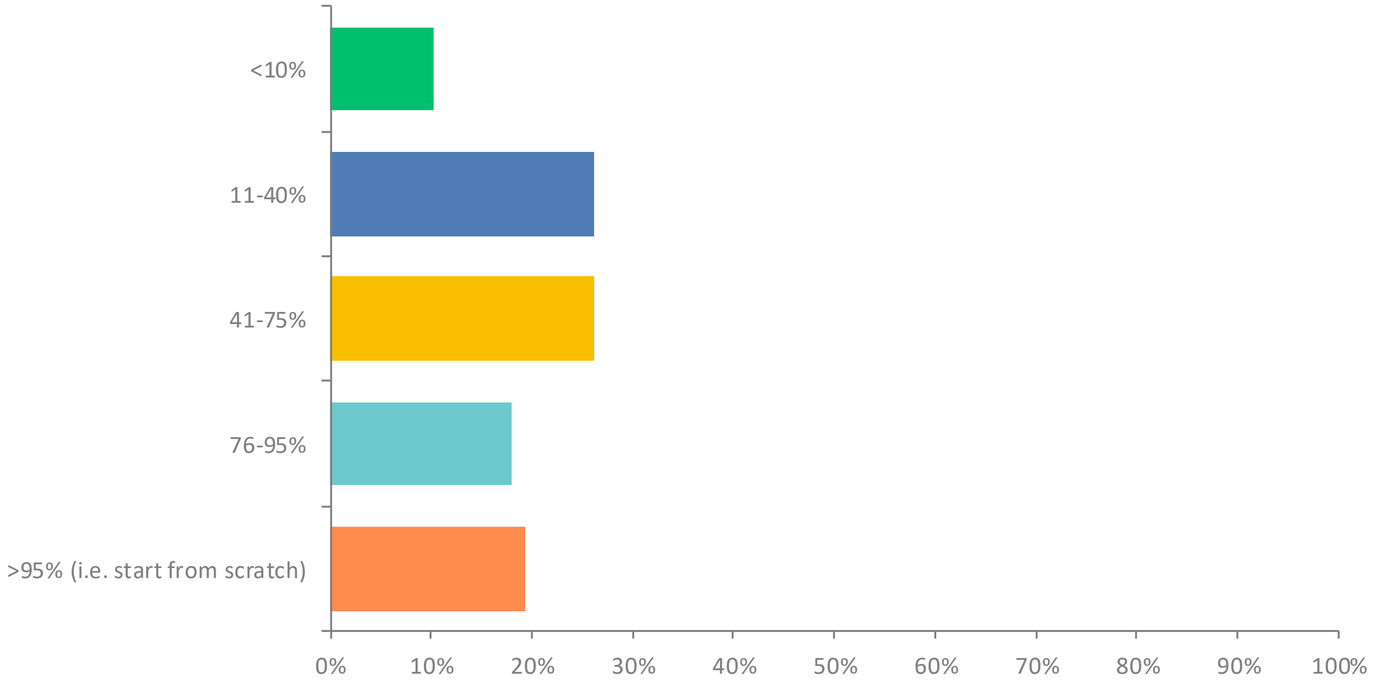
It is encouraging that over 40% have had over 3 days of training in this. But judging from some of the other replies, I'm not sure that this learning is being applied, although the reasons for this may not be technical ones (see Question 10).

Why should automators learn about this? To minimise maintenance, the testware should be structured to be modular and robust. Ideally the automated tests should be separated from whatever tool is currently being used with tool-specific scripts at a minimum. This enables both easier maintenance when the software changes, but also ease of migration to a different tool or platform. (Two test automation patterns apply here: Abstraction Levels and Testware Architecture - see link to the patterns at the end of this PDF.)

There should also be a good interface to the automation for the non-technical (i.e. non-coding) testers, so that they can easily write and run tests in an automated form. This will help to make the automation more widely used, thereby gaining increased benefit. It seems to me that many commercial tools now have quite a good interface for testers; however, this comes at a cost of then being locked in to that vendor and possibly losing the lower level of abstraction. Is it the other way around for open source tools? Do they have a reasonable way to achieve abstraction at the technical level (with the right investment) but the user interface may be lacking? At this point I don't know, but would like to.

QUESTION 7

If you had to change your automation platform, how much of the testware would need to be changed, in your view?



ANSWER CHOICES	RESPONSES	
<10%	10.19%	21
11-40%	26.21%	54
41-75%	26.21%	54
76-95%	17.96%	37
>95% (i.e. start from scratch)	19.42%	40
TOTAL		206

Dorothy's Thoughts

This was one of the most interesting responses for me, and this tells me about that bottom layer of abstraction, the technical separation of the tool from the automated testware. This should be part of learning how to do automation well by applying good programming practices to automation code.

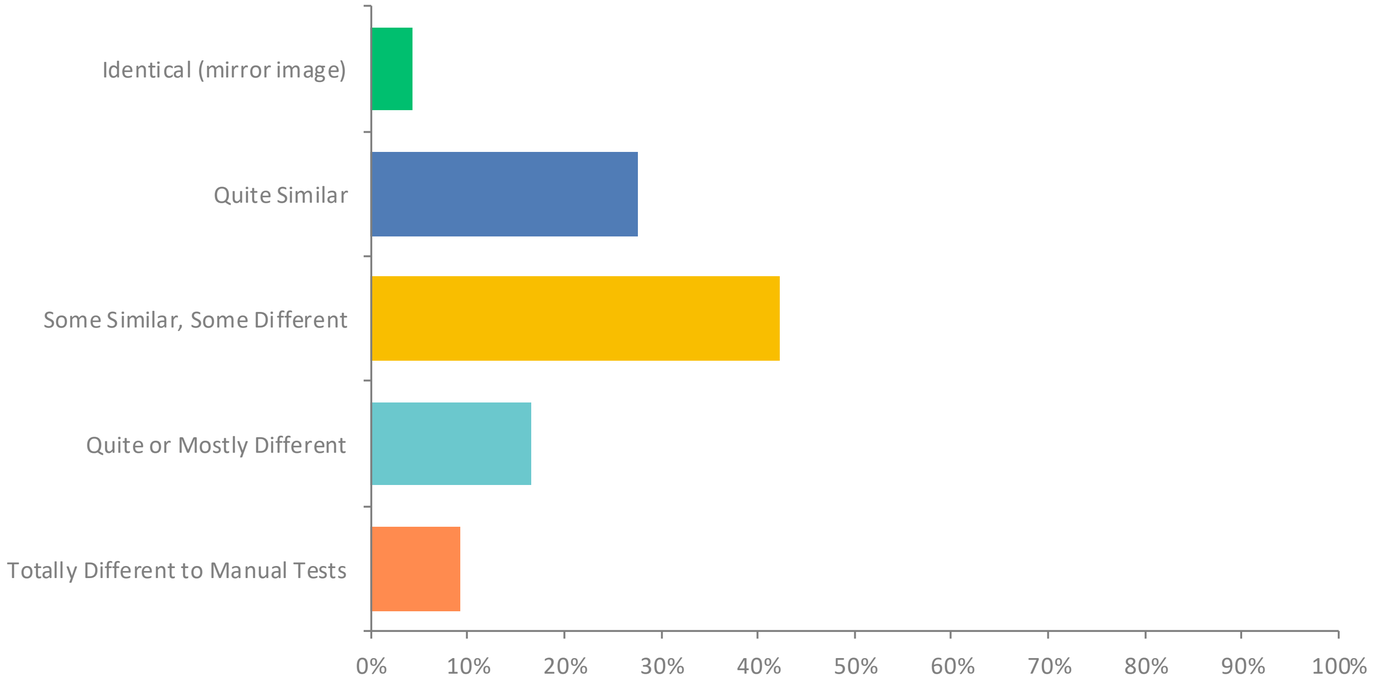
Sadly, only 10% of respondents seem to have a good technical abstraction level, which means they would preserve their investment in the automated tests even when they have to change to using a different tool or platform. I suspect that they probably have much less maintenance to do when the software they are testing changes also.

Almost 40% would lose 75% of their investment in the automated tests, and 19% would end up throwing it all away and starting from scratch - this is wasteful and not good automation.

It may be that many of the respondents would have liked to put good design principles into practice, but due to misunderstandings and unrealistic expectations at the management level, they were prevented from making this investment - it does take time and effort up front to construct good automation - but the payback later on significantly outweighs this investment. (See Question 10)

QUESTION 8

How similar is the structure of the automated tests to the structure of the equivalent manual tests?



ANSWER CHOICES	RESPONSES	
Identical (mirror image)	4.37%	9
Quite Similar	27.67%	57
Some Similar, Some Different	42.23%	87
Quite or Mostly Different	16.50%	34
Totally Different to Manual Tests	9.22%	19
TOTAL		206

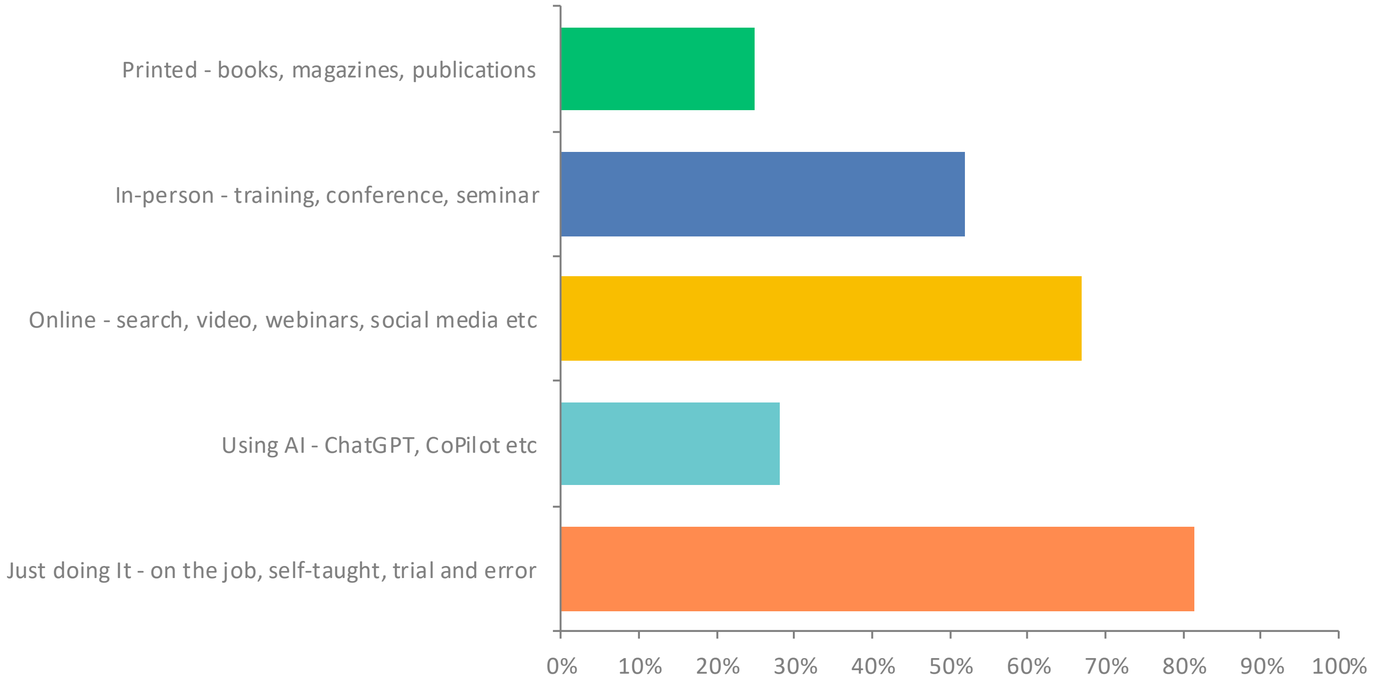
Dorothy's Thoughts

This response also leads me to believe that people are not getting that lower level of abstraction right in their testware architecture. An automated test should have a different structure than its equivalent manual test. A manual test consists of steps taken by a human being in running that test. With 100 manual tests, you may well be doing the same thing many times (e.g. log-in) before coming to the unique part of the test. When this test is automated, the common, duplicated actions should be stripped out and made into a script that is called as part of the set-up for the test. Chris Loder reports that one manual test may consist of 3 to 10 automated scripts as a “rule of thumb”, but may be as high as 30. One automated script may be used by dozens or even hundreds of tests. See Chris’s webinar on automation (referenced at the end of this PDF).

Automated scripts should be different to their manual equivalent tests.

QUESTION 9

How do you find out about automated testing?
(Tick all that apply)



ANSWER CHOICES	RESPONSES	
Printed - books, magazines, publications	24.76%	51
In-person - training, conference, seminar	51.94%	107
Online - search, video, webinars, social media etc	66.99%	138
Using AI - ChatGPT, CoPilot etc	28.16%	58
Just doing It - on the job, self-taught, trial and error	81.55%	168
TOTAL		522

Dorothy's Thoughts

I wondered how people learned about automation. Having written 3 books on the subject, I wonder if people are still reading books! It seems that some people are still reading printed material, but most people learn through trial and error, or from their colleagues at work, or by just doing it as best they can. However, it looks like many people also are getting information from more than one source, and no surprise that the internet is the 2nd most popular choice. Are people getting the right training in automation? Not everything you find on the internet is of the best quality!

It is interesting that over a quarter of respondents are using AI to find out about automation. I wonder how well that is working? I suspect that AI can be useful, but only if it is used wisely!

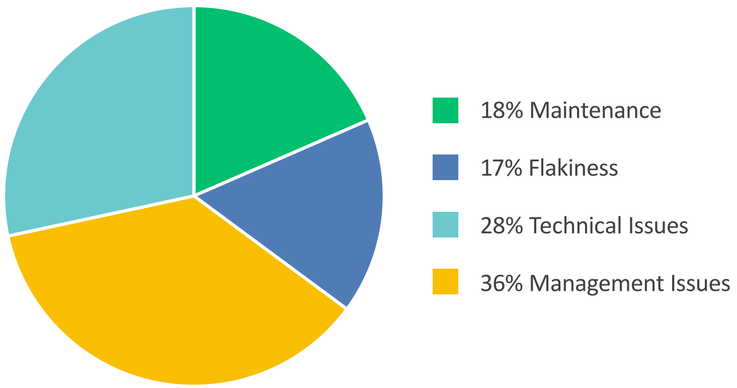
QUESTION 10

What is your worst problem with automated testing?

(brief explanation, or any other comments, benefits achieved, etc.)

This was the free text field of the survey, and there were a lot of very interesting replies. There were some people who are clearly doing really well. One reported using their home-built tool for over 10 years, simulating thousands of users and with clear and traceable reports. One other person said that they really appreciate the automated tests - they prevented lots of bugs going to production, plus they get a better understanding of the software when writing automated tests.

But I did ask about problems, and got plenty of those. I have categorised them into four areas, which covered most of the comments: Maintenance, Flakiness, Management Issues, and Technical Issues (See chart below).



Maintenance

I was expecting maintenance to be one of the most popular problems, but it was not - only 18% of people named maintenance as their worst problem (so this is still a significant problem). Examples: constantly changing applications, frequent updates required, test automation requires work every time the target software requires work.

Flakiness

Flakiness of the automated tests was quoted by a similar number of people: 17%. People mentioned reliability of tests, tests that sometimes pass and sometimes fail, and false failures (and presumably false passes as well).

Technical Issues

Technical issues were mentioned by 28% of the respondents. This included data problems, environment problems, dynamic IDs, lack of integration, setting the system state, infrastructure, networks and coding. The sub-topic most mentioned was about data issues.

Management Issues

The most popular problem was management issues, mentioned by 36% of respondents. These comments fell into two sub-groups; one half mentioned issues such as lack of time or resources, attitudes, lack of training, culture, and knowing what to automate.

The other half mentioned unrealistic expectations from management as their worst problem. Some of those comments: thinking that automation is just about buying the right tool, they want it yesterday and they think it should just take a few minutes, the company pushes for automation but without adequate time and resources, expecting magic and no maintenance - I've seen this twice (so far).

The importance of management in good automation is often under-estimated. If managers are not well-informed about the actual costs and investment needed for good long-term solutions, budget and time pressures will ensure that automation will not fulfil its potential. A short-term view, expecting an instant solution, i.e. a miracle, leads to lack of training, lack of needed time to construct a lasting solution, and ultimately costs a lot more.

How can we educate managers? Some of this responsibility lies with us, the testers - learn how to better communicate about automation with higher management.

QUESTION 10 (Continued)

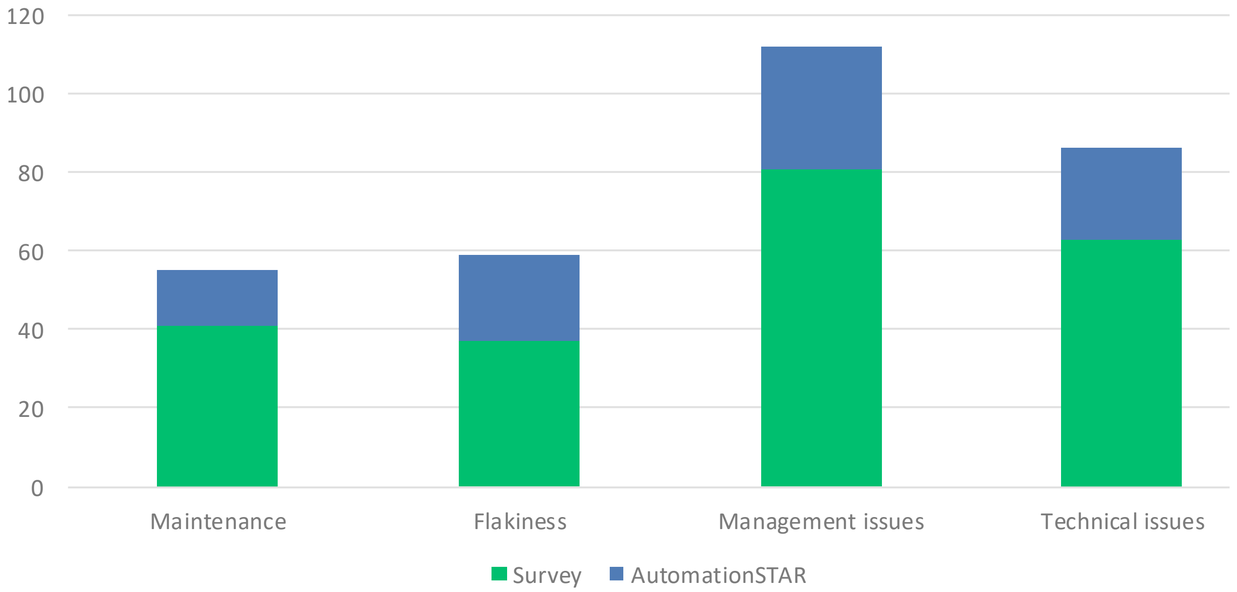
What is your worst problem with automated testing?

(brief explanation, or any other comments, benefits achieved, etc.)

Additional Data

At the AutomationSTAR conference (9th-10th October 2024 in Vienna), I took a poll on the Whova app that was used by the conference, asking this same question.

Analysing the results gave me slightly different numbers, but again, management issues were the most often mentioned. This summarises my categorisation of the replies from both:



Area	Survey		AutomationSTAR		Totals	
	Number	Percent	Number	Percent	Number	Percent
Maintenance	41	18.5%	14	15.6%	55	17.6%
Flakiness	37	16.7%	22	24.4%	59	18.9%
Management Issues	81	36.5%	31	34.4%	112	35.9%
Technical issues	63	28.4%	23	25.6%	86	27.6%
Total	222		90		312	

Conclusions

The purpose of this survey was to get an idea of what people are doing in automation at the present. Are people better informed about automation now than they used to be? Are people doing automation well these days? I was particularly interested in issues around testware architecture and abstraction levels.

The issues raised in this survey seem to be very similar to the issues of 20 years ago. It is particularly sad to see that a fifth of respondents were struggling with unrealistic expectations from their managers, stakeholders and customers. How can we do better at getting the message across that automation takes effort and is not a magic solution?

I found it interesting that 80% of people are using open source tools. But how well are these tools being used? The layers of abstraction essential for ease of maintenance seem to be missing for many of those surveyed. If nearly 40% of the respondents have to throw away more than 75% of their automated testware when changing to a new platform, their investment in automated tests (as opposed to tool use) has been wasted.

It is disappointing to find a very low level of training in automation, even in the use of the tool itself. Training is often the first cut made to save money, but the additional cost in trial and error, time spent working sub-optimally, possibly never achieving the potential in using the tool may far outweigh the cost of a few days of training. Skimping on training is not saving money, it's wasting time (and time is money, right?).

This survey is inevitably a bit negatively biased, but I was asking about problems! However, what I found at the AutomationStar conference was a community full of enthusiasm about automation and eager to learn, with lots of great advice about doing automation well, so I am optimistic that the current crop of problems can and will be overcome.

More Resources for Test Automation

Test Automation Patterns: ***www.TestAutomationPatterns.org***
See particularly the patterns 'Testware Architecture' and 'Abstraction Levels' (among 80+ others)

Isabel Evans' research into ***test tools*** and the specific figures on ***training.***

Chris Loder's webinar on starting ***doing automation well.***

Chapter 29 from our book (with Mark Fewster)
Experiences of Test Automation.

AutomationSTAR

www.automationstar.com

