LogiGear

Today's Regression Automation Challenge for CI/CD

Michael Hackett
Senior VP at LogiGear Corporation

1

TODAY'S AGENDA

The Modern CD Regression Challenge

- It's not like it used to be…the Problem
- What is Lean?
- What is Regression?
- What is Continuous Delivery?

- Automation Topic Discussions
- Solutions

- Summary
- Q&A

Everybody wants to do CD!

The Excitement:
It all sounds great. The benefits are unmistakable. Everybody wants to use the cool new tools…

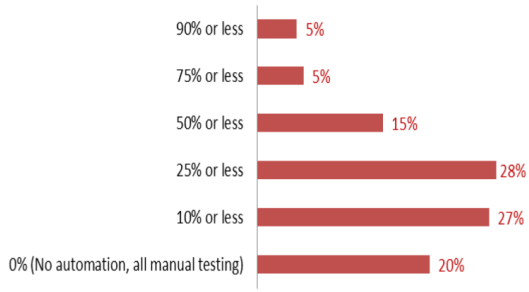
The Problems:

- "But we have a giant monolithic regression suite that takes from overnight to 5 days to run."
- "Not all of our important tests are automated—we have important manual regression."
- "No one else can run our automation."

- But—the Team wants your pipeline tool to run all our test automation in minutes and give you an easy yes/no result to promote code or not? CD is not going to happen.

Where are you on the spectrum?

What percentage of your tests are automated?



Taken from the LogiGear 2017 State of Software Testing Survey series:
http://www.logigear.com/magazine/survey/survey-results-testing-essentials/

The Modern Regression Challenge in CD

- Test Automation is not optional.
- We also know the "automate everything" myth is just that—a myth.

Current state

- There are some organizations with 100s of 1000s of tests constantly running on all kinds of on-prem VMs.
- There are many, many more running thousands or 10s of 1000s of tests.
- There are also many organizations running 100s.
- There are still a surprisingly high number of organizations with little to no automation.

Every test team— regardless of where you are on the automation spectrum will have hurdles moving to CD.

The Modern Regression Challenge in CD

Other problems that will block you from CD:

- We have a lot of false +/-. The analysis takes a long time.
- We have a painful set up the data before a run.
- The automation maintenance time is high—sometimes we let fails slide since we know the tests and the change list…
- The regression automation suite is giant, so we have a pass rate**.

Any hope of Continuous Testing or Pipeline Automation makes these current states unsustainable.

It's not like it used to be…

How did we get here?

- Too much focus on UI test automation, or, that was the only interface even considered for automation
- *Every test* must be automated.
- Some teams automated tests because they could! Regardless of the value of the test.
- Some mythical number put out by someone who thought they knew something:
  "You have to automate 75%."
- Or, there is some Definition of Done criteria like—every acceptance criteria must have an automated test.

Many tests may have been automated which, in the long term, were not the best tests.

Our automation must change.

For CD we need:

- Fast running—as in minutes or hours, not days.
- Immediate, easy feedback.
- Easy to run.
- Easy to troubleshoot.
- Easy to analyze.
- Meaningful test automation.

What's the solution?

Even with using Sauce Labs**, this makes running tests easy and takes care of the *machine* part—but not your automation.

Have a great definition of Regression!

Do you run:

- *All* tests?
- P1? P1 and P2s?...
- Happy Path?
- Most common paths?
- Bugs found?
- Based on change set/changed area?
- Core or main functionality?
- Smoke Tests?

Regression is more complex than you think.

Description	Time to Execute	Coverage	Risk
Full			
Core Functionality			
Main Paths			
Most Common Paths			
Bug			
Smoke Test			

Does everyone on your team understand the differences here?

What's the solution?

- Teams must rethink WHAT they automate and WHEN to run the tests.
- Different suites will have different *Coverage* and *Risk*.
- Move to multiple smaller suites run at different times.

Lean/Lean Software Development

The foundation of CD:
https://en.wikipedia.org/wiki/Lean_software_development

7 Lean principles:

- Eliminate waste
- Amplify learning
- Decide as late as possible (JIT)
- Deliver as fast as possible
- Empower the team
- Build integrity in/Quality at every step
- See the whole

Lean/Lean Software Development

Eliminate Waste.

- Cut what's not needed.
- Stay focused.
- Run the essential, no extras.
- These will be tough decisions and may increase risk.

Lean/Lean Software Development

Deliver as Fast as Possible.

- Immediate feedback.
- 5 days to run automation is not immediate. We need to *fail fast*.
- Make sure it is giving a fast, consistent result.
- Many CD organizations build multiple times a day. What will you run those builds against?

Lean/Lean Software Development

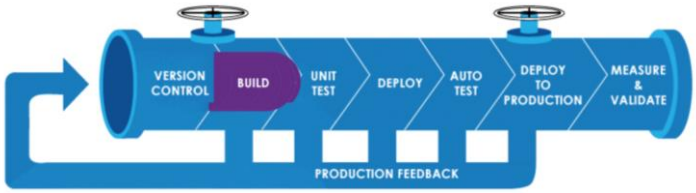
Quality at Every Step has a specific CD meaning.

- Do something **small**: test it. See that it works, be focused in regression.
- No big block testing at the end! Test at every step.
- Don't think in overall terms of test importance but low-level focus *how important is this test at a certain point of development?*

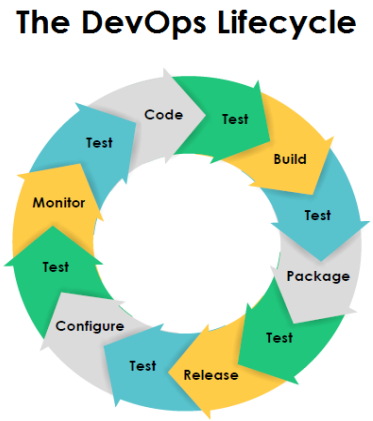
What is CD?

Continuous Delivery (CD) – CD is a software engineering approach in which teams produce software in short cycles, ensuring that the software can be reliably released at any time. It aims at building, testing, and releasing software faster and more frequently. The approach helps reduce the cost, time, and risk of delivering changes by allowing for more incremental updates to applications in production. A straightforward and repeatable deployment process is important for continuous delivery.

Source: https://en.wikipedia.org/wiki/Continuous_delivery



Lean/Lean Software Development



Automation Strategy

Automation strategies:

- Many teams have never had an automation *strategy*.
- Some teams automated tests *because they could*, not because they were important.
- This led to—over time—well-meaning but, bloated and slow suites.
- Some teams have never taken much time to clean out tests.
- Retiring tests makes many people feel nervous. Its added risk.

Automation Strategy

Bloated & Slow vs. Lean & Mean

- For teams with automation goal of *more, more, more,* shift to *better, smarter, and more focused.*
- Re-focus by suite.
- Retire old, duplicate, inefficient tests.
- Optimize tests: go for boundary/edge cases rather than simple, *happy path*.
- Trace, tag and prioritize tests to cut down at run time.

Automation Style

There is great discussion today on automation strategy.

Atomic, decoupled, small focus tests
vs.
Workflow, spaghetti, stringy tests…but are *real* user scenarios

Atomic vs Spaghetti

- "The atomic isolated function tests are easier to write, faster to run, easier to debug and analyze."
- "Why not do all real user scenarios/workflows for your automation? Its what our users do. " Primarily because these workflows are harder to write, break more and are harder to analyze.

Atomic vs Spaghetti

- Both are essential!
- They show different things.
- The distinction is a matter of scale and balance.
- How much of one type vs how much of the other.
- Most teams will have more atomic tests but, you cannot leave out scenarios.

CD Automation Discussion

Good Frameworks can reduce production and maintenance time—they better!

- But this does not impact having too many tests, old tests, false +/-
- Or analysis time
- Or atomic vs workflow tests.
- Merely a good framework will not solve these problems.
- A better test design will solve it.

CD Automation Challenge Summary

- Define regression.
- Define your automation goals and strategy.
- Design many suites.
- Communicate coverage and risk.

Solutions Mindset

Move from monolith to agile suites: imagine you having multiple automated suites:

- Smoke tests.
- Integration tests.
- Mock API tests.
- Live API tests.
- Database tests.
- UI tests.
- Browser compatibility tests.
- Mobile responsiveness tests.

Lots of Lean and Mean automated suites is much better than a monolith of bloated and slow.

Solutions

Make smarter automation choices about what to automate at what level-Unit, API, UI. Also use this to cut redundancy.

Flipping the Automation Triangle

Solutions

Re-think test automation strategy. How to make choices or what tests to run.

- Prioritize tests: this is often still not successful. Too many, they get old. Choice is on priority not pipeline-focus area.
- Traceability and tagging: pull tests only associated with certain chunks of code. Some tool suites do not easily support this.
- Change set choices: if traced or machine managed- great. If it's a guess or hope—trouble.
- Risk-based choices: OK if function, or area focused. If risk is similar to priority—same problems.

Solutions: Very little to no Automation

If you are one of those many teams with no or very little automation:

- Define regression.
- Define goals for automation.
- Start now and do it right!
- Get help.
- Start small. Smoke tests.
- Data Driven testing. Small number of scripts with lots of data.
- A few E2E or workflow tests.
- Tag, trace, and prioritize well!

You are moving to CD: Overhaul

- Be ruthless, merciless cutting tests to move from *Bloated and Slow* to *Lean and Mean*.
- Use traceability and tagging to sort through existing tests.
- Examine isolated function tests to remove if already unit tested or it's a boring, low value test.
- Do E2E-type workflow tests—but keep this set small.

You are **not** moving to CD: Overhaul

In addition to all the other ideas…

- Be careful not to over-automate low value tests.
- Define an economic strategy.
- Focus strategy on lower level tests first (triangle).

Summary

Get a better grip on your automated test suite!

- What success will look like: define automation success, and create strategy to get there
- Be selective: make smarter Automation choices
- Spring-cleaning: regularly clean out stale tests.
- Watch your diet: keep test suites Lean and Mean.

Summary (cont.)

- Know your tests: use Traceability, Tagging, and Priority!
- Small is beautiful: smaller, focused sets of automated tests targeting smaller sets of objectives.
- Keep automation simple: but not the tests
- Author not included: design your suites that can be run, analyzed and troubleshot by a tool or other people.

Contact

▷Email me at MichaelH@logigear.com

▷Connect with me on LinkedIn: https://www.linkedin.com/in/michael-hackett-a0575b2

▷Tweet me @logigear!

Thank You!

▷Thank you for joining us today. To learn more about LogiGear and our continuous testing services visit: http://www.logigear.com/solutions/continuous-testing-solution.html

▷*Visit LogiGear Magazine and get the latest copy—it's all about DevOps!*