LogiGear

Today's Regression Automation Challenge for Continuous Delivery

Michael Hackett
Senior VP at LogiGear Corporation

TODAY'S AGENDA

- The Modern CD Regression Challenge

- It's not like it used to be… the Problem
- What actually 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…

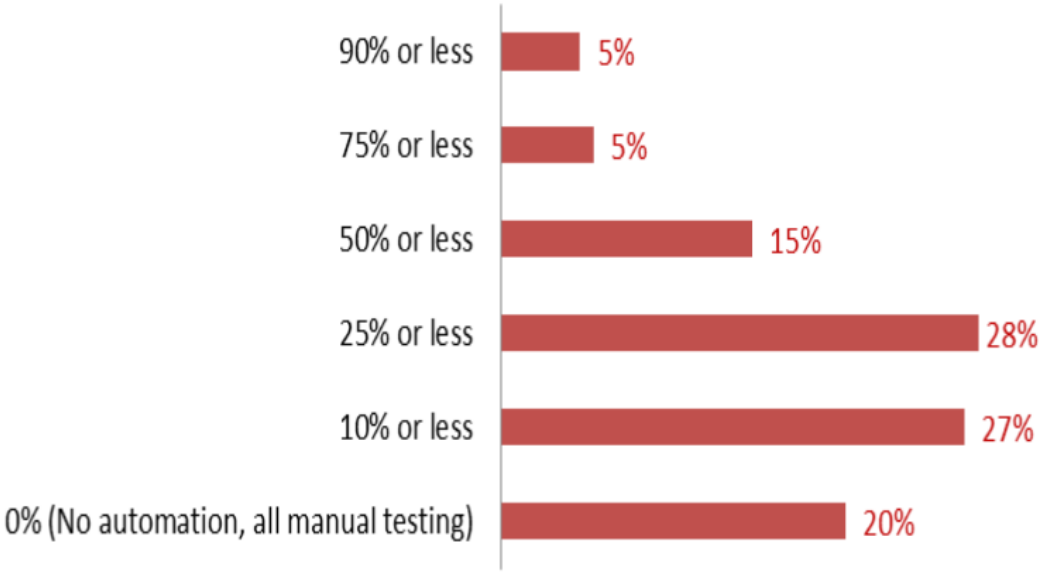
The Problems:

- "But we have a giant monolithic regression suite that takes from overnight to 5 days to run."
- "Not all of our important test 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 thousands of tests constantly running on all kinds of VMs.
- There are many, many more running thousands or 10s of thousands 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 to set up the data before a run
- We love our tool- but it has some bugs, our code has some bugs, the scripts have some issues…it's a bit of a mess.
- The automation maintenance time is high- sometimes we let fails slide since we know the tests and the code 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?

- *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.

It's not like it used to be…

- Also, some teams have been very aggressive about automation for a long time- doing a great job automating- and have thousands or tens of thousands of tests, and have never taken much time to clean out tests.
- Retiring tests makes many people feel nervous.
- But, now we need to make a shift and have *everything* run in minutes, not overnight or a week.
- There are tough decisions to make.

It's not like it used to be…

How do you deal with fast:

- Rollbacks?
- Hot fix?
- Swapping containers?
- Small change set going live?

- Re-run a giant suite for 5 days? No.
- Some teams have this handled. Most do not.

It's not like it used to be

For CD we need:

- Fast running
- Immediate, easy feedback
- Easy to run
- Easy to troubleshoot
- Easy to analyze
- Meaningful test automation

What's the solution?

- Teams must rethink WHAT they automate and WHEN to run the tests.
- Let's start with some basic definition and understanding of processes and goals.

What is Regression?

REGRESSION

1: the act or an instance of regressing

2: A trend or shift toward a lower or less perfect state:

c : reversion to an earlier mental or behavioral level

https://www.merriam-webster.com/dictionary/regression

What is Regression?

- Regress is simply to go back to a previous state or do again .
- Every test you re-run again is actually a regression test or check against a previous state.

- For this Regression Test Suite, what tests do we go back and choose to verify against the previous state- to show consistency- to show we are not regressing?

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?

Have a great definition of Regression!

What is the Goal of automation?

- Each of these is a different sets of tests, covering different aspects of the system and giving different results.
- We need to start thinking of regression as multiple, different suites to be run at different times.

Lean/Lean Software Development

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

7 Lean principles:

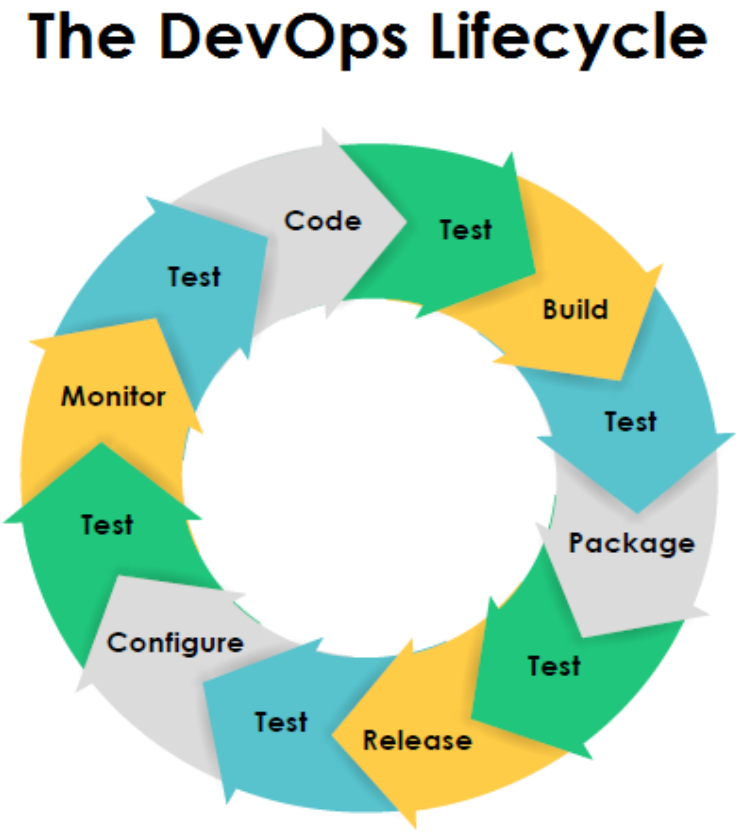
- Eliminate waste
- Amplify learning
- Decide as late as possible (JIT)
- Deliver as fast as possible
- Empower the team
- Build integrity in/Quality at every step
- See the whole

Lean/Lean Software Development

Quality at Every Step has a specific CD meaning.

- Do something: test it. See that it works, verify it did not break anything.
- Make sure it is giving a consistent result and not going backward, regressing.
- No big block testing at the end! Test at every step.

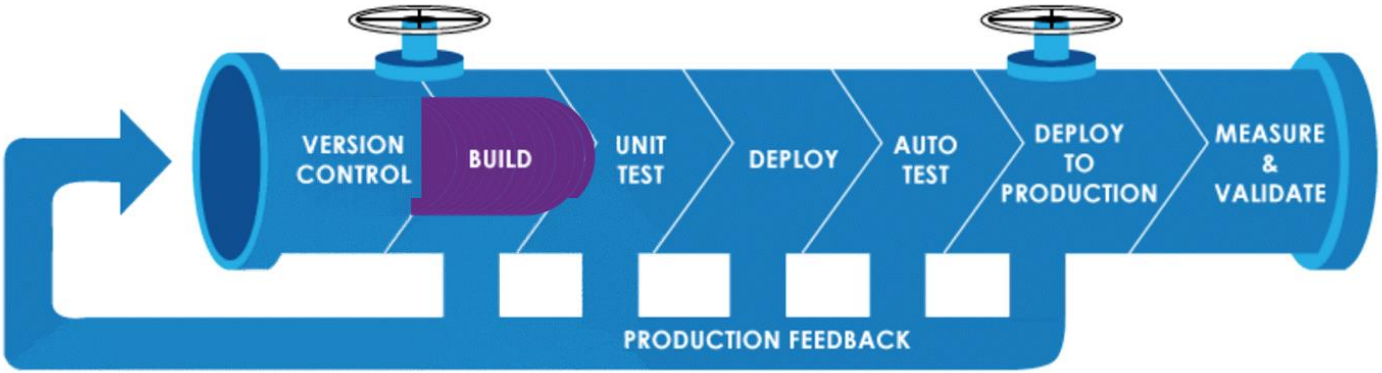
Lean/Lean Software Development



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



What is CD?

Using tools like:

- Jenkins
- Team City
- Bamboo
- XL Deploy
- IBM Urban Code

The pipeline automation tool will go grab and run the automated tests-no human contact-and promote or not the product to the next environment based on the quick easy results. The tool won't wait overnight.

What is CD?

- In CD where immediate feedback is the goal, and the idea of re-running *everything* – or however you define regression- at every step is too much.
- Choices have to be made for tests to find bugs, show consistency, validate whatever step you are on and give immediate feedback.
- This is not easy.

What is CD?

- Test automation can no longer slow teams down, break often, or give ambiguous results.
- Test automation must be an asset to the Dev team that speeds deployment and also gives fast, immediate feedback and confidence in the system.

Automation Strategy

Automation strategies:

- Many teams have never had an automation *strategy*.
- Remember from earlier- Some teams automated certain tests *because they could*, not because they were important.
- This lead to, over time, well-meaning but Bloated and Slow suites.

Automation Strategy

We know:

- We need automation. A lot!
- CD depends on automation.
- We need automated tests at every step.

What do we do?

- Many, focused, well-defined smaller suites to be run only at exact times in exact environments.
- When you make a change- focus tests only on that change.

Automation Style

There is great discussion and problem today on automation approach.

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

Atomic vs Spaghetti

"The atomic isolated function tests should all be unit tests. "

"Why not do all real user scenarios/workflows for your automation? "

Primarily because these workflows are harder to write, break more and are harder to analyze.

Atomic vs Spaghetti

Before we get too nervous:

- Remember test design.
- There are many different reasons to test. Our test design needs to reflect that.

Atomic

- There are tests that show a single function, run through the UI will give us the expected result.

Workflows: Logic

There are also very different tests we design that show logic.

- To show that one function or system integrates with another function or system or API or device correctly. That is a different kind and style test.
- First a user does this, then they go do the next thing, then another…
- These tests find a different kind of error than running the isolated functions!

Atomic vs Spaghetti

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

Atomic vs Spaghetti

The solution here is going to need a clear explicit Test Strategy and Automation Strategy:

- What functions have unit tests? Re-running those is enough. You don't need UI-driven function tests there.
- What functions do you need UI-driven tests for?
- What are key workflows to validate?

Balance these.

Automation Strategy

Bloated & Slow

vs.

Lean & Mean

- It's about time we have this discussion regardless of CD.
- For teams who's automation goal was more more more we need a shift to better smarter more focused.

Automation Strategy

Bloated & Slow
vs.
Lean & Mean

- Re-focus goal of the suite
- Retire old tests
- Optimize tests: go for boundaries instead of Happy Path
- Tag, trace and prioritize tests to cut down at run time.

CD Automation Discussion

Having someone else run, debug, analyze your results.

- Many teams are controlling of their tests. The tests, data or tool may be problematic to run. This does not work in CD.
- In CD the running gets passed to a tool.
- Many teams have had to re-engineer large numbers of tests or change tools or simply have a significant engineering project to have automatic running of suites.
- This is critical for CD.
- Smaller focused and well-designed tests are what you need.

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 Discussion

Lean and Mean automation:

- Running tests against multiple systems, devices, platforms
- Running tests is no longer free.
  …costs to run against devices, cost of running in the cloud
- Smaller, focused suites benefit here also.

CD Automation Discussion

- Define regression.
- Define your automation goals.
- Design many suites.

Solutions Mindset

- Automation, like many applications, needs to move from a giant monolith to an agile, nimble variety of suites used at different time for different goals.
- Don't think in terms of *overall* how important but low-level focus *how important is this test at a certain point of development?*

Solutions Mindset

It is easy to quickly 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.

Lean and Mean Automated Suites

- If, for example, one step in deployment automation is to hook in live databases instead of smaller, faster or mocked databases used for testing earlier in the pipeline, the automation suite run at that step should only be about that database switch. A "full regression"' is too slow and not necessary.

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- 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.

- Start now and do it right!
- Define regression
- Define goals for automation
- 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 others 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 would look like—define automation success, and create strategy to get there
- Be selective--Make smarter Automation choices
- Spring-cleaning--Regularly cleaning out stale tests.
- Watch your diet--Keep test suites Lean and Mean.

Summary

- 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.


Q
A

Stay in Touch!

- 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!*

LogiGear USA - Headquarters
4100 E 3rd Ave, Suite 150
Foster City, CA 94403
Tel: +1650 572 1400
Fax: +1650 572 2822

LogiGear USA - Houston
10777 Westheimer Suite 1000
Houston, TX 77042
Tel: +1 650 572 1400 ext. 380
Fax: +1 281 288 9088

LogiGear USA - Seattle
2225 N 56th St
Seattle, WA 98103
Tel: +1 650 572 1400
Fax: +1 650 572 2822

LogiGear Viet Nam - Headquarters
1A Phan Xich Long, Ward 2
Phu Nhuan District
Ho Chi Minh City
Tel: +84 839 954 072
Fax: +84 839 954 076

LogiGear Viet Nam - Da Nang City
346 Street 2/9
Hai Chau District
Da Nang City
Tel: +84 511 365 533
Fax: +84 839 954 076

www.logigear.com | www.testarchitect.com