A Holistic Approach to Testing in Continuous Delivery

Lisa Crispin

With material from Abby Bangser, Ashley Hunsberger, Lisi Hocke, Janet Gregory, & more

Copyright 2020 Lisa Crispin
A little about me

Co-founder, Agile Testing Fellowship
Testing consultant & trainer
lisa@lisacrispin.com
https://lisacrispin.com
Today I’m talking about:

● Building confidence for continuous delivery/deployment (CD)
● Guiding conversations about risk & test coverage
● Quality – a whole team responsibility
Some journeys go wrong

- Slow feedback loops
- Regression failures
- Unexpected impacts
- Technical & testing debt
One example from my experience

- Team embraced XP practices – TDD, refactoring, pairing, continuous integration, ...
- Thousands of automated regression tests at every level
- Reliable deployment pipelines, app in cloud, blue/green deploy

But...

- Too few testers
- Still had manual release regression checklist
- No time for sufficient exploratory testing
What gets in your team’s way?

Stop and think for a minute. What’s the biggest obstacle for your team being successful with continuous delivery?

Photo by Matthew Hamilton on Unsplash
Making frequent small changes confidently takes:

- Collaboration
- Continuous improvement
- Continuous learning
Holistic Testing

- Identify risks
- Test assumptions
- Create testable stories

- ATDD / BDD
- Example mapping
- Prototypes
- Determine what to observe or monitor

- Test the ideas
  - Determine value

- Test infrastructure
  - Run automated tests
  - Test the pipeline

- Test quality attributes
- Test the system

- Test in production
  - Use feature toggles or blue/green env

- Automate tests
  - Instrument the code

- Test stories and features

- Hypothesize and adapt

- How do customers use the product
- Monitor for warnings and errors

Janet Gregory
https://janetgregory.ca/testing-from-a-holistic-point-of-view/
Discover and plan

- Identify risks
- Test assumptions
- Create testable stories
- Test the ideas
- Determine value

Impact mapping
- Goal
- Actor/Stakeholder
- Impact
- Deliverable
- Who?
- How?
- What
- Why?
- Who?
- How?
- What
- Who?
- How?
- What

Risk assessment
- Probability
- Impact
- High risk
- Medium risk
- Low risk
Test early
– test our understanding

- Ask questions
- Uncover hidden assumptions
- Clarify needs (ATDD / BDD)
- Think about testing first
- Give tests to the programmers – before coding happens
Testing activities while we build

- TDD (test-driven development)
- Code analysis
- “show me”
- Exploratory testing
- Test automation
- User acceptance testing
The shift right half of the infinity loop

- Test infrastructure
- Run automated tests
- Test the pipeline
- Test quality attributes
- Test the system

- Test in production
- Use feature toggles or blue/green env
Observing and Learning

- Hypothesize and adapt
- How do customers use the product
- Monitor for warnings and errors
In which stages do you currently get involved?

What relationships could you build to participate in other parts of the loop?
Get everyone engaged
Guiding conversations with visuals
Visualize your pipeline, optimize feedback

Faster feedback

More confidence
Do your automated test suites give you confidence?

- Flaky tests?
- Poor coverage?
- Hard to diagnose?
- Hard to maintain?
**Why**
What business question am I trying to answer with this suite? What risk does this suite mitigate?

**Dependencies**
What systems or tools must be functional for this suite to run successfully?

**Constraints**
What has prevented us from implementing this suite in an ideal way? What are our known workarounds?

**Pipelining / Execution**
Is the suite part of a pipeline? When is it triggered? How often does it run? Is it gated?

**Data**
Do we mock, query, inject? How is test data setup/managed?

---

**Engagement and Failure Response**
Who created the suite? Who contributes to it now? Who is not involved but should be? In the event of a test failure, who addresses failures and how?

**Maintainability**
What is the code review process? What documentation exists?

**Effectiveness**
How do we know the suite is effective? What is it finding? What is it preventing?

---

[https://github.com/ahunsberger/TestSuiteDesign](https://github.com/ahunsberger/TestSuiteDesign) - Ashley Hunsberger
Some of my favorite Test Canvas questions

- What information should each suite provide? To whom? How?
- How will the team know about test failures? Who is responsible for looking into those?
- Do you pair on test automation, or do test code reviews?

This Photo by Unknown Author is licensed under CC BY-NC
Mitigating risks
Talking about risk

Risk Storming
(https://riskstormingonline.com)

Mind maps

Traditional risk analysis

@lisacrispin
Does your team know all the possible risks?

- Customers behave in unexpected ways
- Infrastructure components may fail
- External systems can impact ours
- ...
Observability

- Ask the questions you didn’t know you’d need to ask - unknowns
- Complex systems fail in complex ways
- With enough information, we can respond quickly

Cindy Sridharan,
https://medium.com/@copyconstruct/testing-in-production-the-safe-way-18ca102d0ef1
Quality – a whole team responsibility
What makes it work?

- Commitment to a level of quality
  - Identifying what’s valuable to customers
  - Bug prevention over bug detection
  - Fast response to prod issues
- Diverse perspectives, skill sets, biases
- Competencies > roles
Cross-discipline pairing, ensemble testing

Picture from Ensemble Programming Guidebook, Maaret Pyhäjärvi

@lisacrispin
We’re humans! (or possibly dragons, donkeys, unicorns...)

Build relationships
- Friendly conversations
- Do food
- Share something useful
- Ask for help

Katrina Clokie has excellent tips in her book, A Practical Guide to Testing in DevOps
Building a quality culture

- Transformative leaders
- Trust and psychological safety
- Whole team “owns” product
- “You build it, you run it”
- Focus on quality, not speed
Principles of CD – Jez Humble & David Farley

- Build quality in
- Work in small batches
- Computers perform repetitive tasks, people solve problems
- Relentlessly pursue continuous improvement
- Everyone is responsible
One small step at a time

- Use retrospectives to identify the biggest impediment
- Design small experiments to make that less bad
Get your team together and talk

How to fit testing activities into continuous delivery/deployment?

How to get the whole team engaged in building quality in, continuously testing?
A few resources

- "Agile Testing for the Whole Team" training course, [https://agiletestingfellow.com](https://agiletestingfellow.com)
- *Continuous Delivery* by Jez Humble and David Farley, [https://continuousdelivery.com](https://continuousdelivery.com)
- *Accelerate* by Dr. Nicole Forsgren, Jez Humble, Gene Kim
- Ashley Hunsberger’s Test Suite Canvas [https://github.com/ahunsberger/TestSuiteDesign](https://github.com/ahunsberger/TestSuiteDesign)
- [https://lisacrispin.com/observability-continuous-delivery-devops-related-resources/](https://lisacrispin.com/observability-continuous-delivery-devops-related-resources/)
- ExploreIt! Elizabeth Hendrickson

@lisacrispin