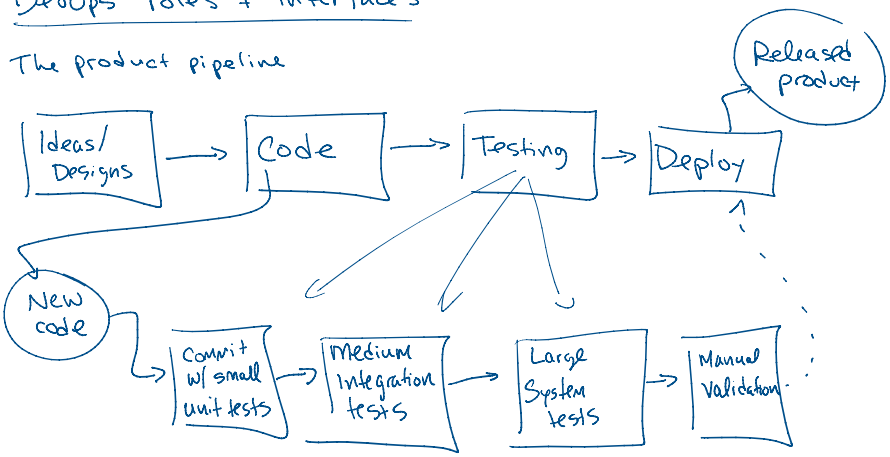
02 History and Motivations for DevOps

Monday, April 11, 2022 10:19 AM

DevOps roles + interfaces

The product pipeline



xUnit

Unit (One Method/class)

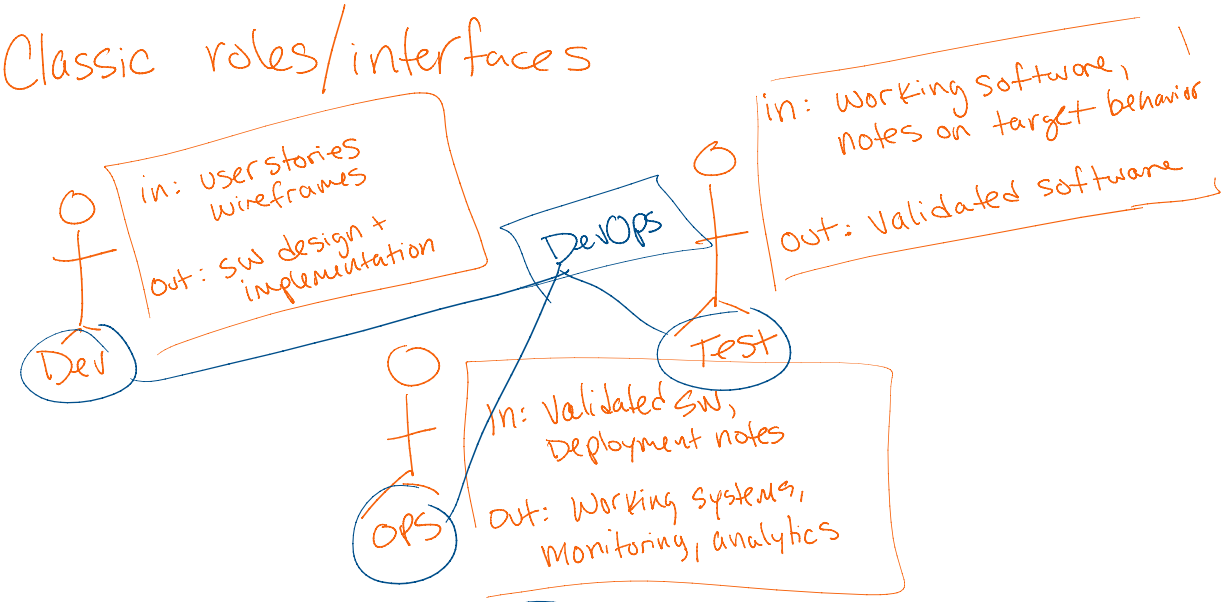
- Runs fast
- High coverage
- Many mocks/doubles
- doesn't test interfaces
- fine resolution

Cucumber

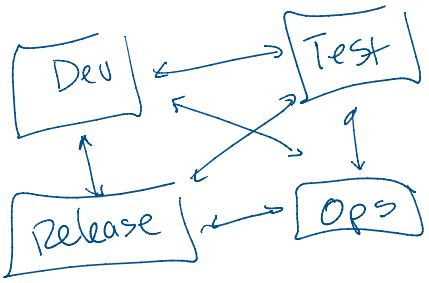
Integration / System

- few mocks
- testing interfaces
- Slow, lower coverage
- Coarse resolution

Classic roles/interfaces



Silo Architecture



Symptoms where DevOps could help

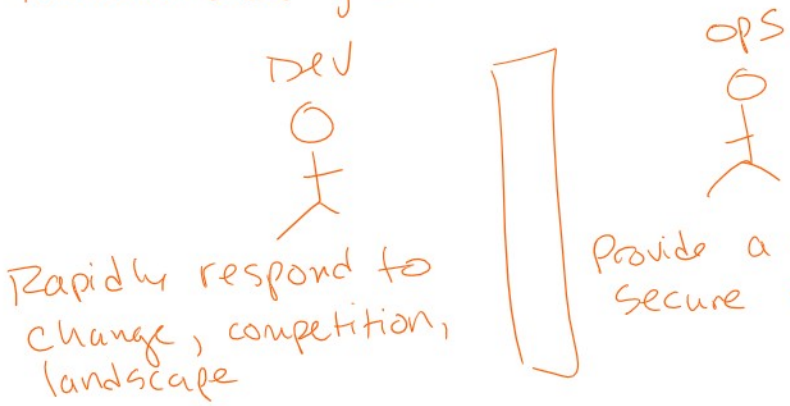
- Developers and IT ops are adversaries
- testing / infosec only happen "later" (too late to correct)
- critical activities have too much

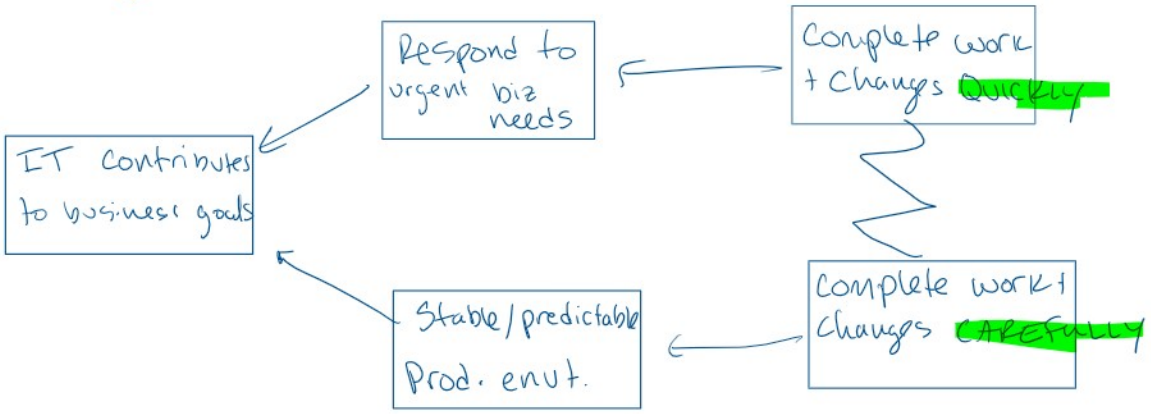
Release ←→ Ops

later (too late to correct)

- critical activities have too much manual effort... too many handoffs...

Core, Chronic conflict:
irreconcilable goals between dev + ops





Similar conflicts in manufacturing:
protect sales vs. reduce costs

- Lean principles helped break this conflict
- Principles behind DevOps are the same
- More than tools; it's cultural

"DevOps" is not a team. It's a culture shift.

Business Value of DevOps:
2013-2016 study: 25K tech professionals
Orgs implementing devops outperformed peers:

- throughput metrics
- 30x more frequent in code change + deploy
  7mx faster in lead time →

- 30x more frequent in code change + deploy
- 200x faster in lead time →
- Reliability metrics
- Production deployments (60x higher success rate)
- faster at restoring service

Convergence of DevOps

- several IT "Movements" in last 20ish years

LEAN movement

1980s to help TPS Toyota Production System
- Kanban boards
- value stream mapping

Two major tenents:
- lead time (raw materials → finished goods) is the best predictor of quality, customer satisfaction, and employee happiness
- small batch sizes help achieve good lead times.

AGILE Movement

2001 Agile manifesto

Velocity Conference movement

2007 → IT ops + web performance
- "10 Deploys per day: Dev and Ops cooperation at Flickr"

AGILE infrastructure movement

2008
- Apply agile to infrastructure, not just application code.

CD - continuous delivery movement

- Deployment "pipelines"
- 2006 ~ continuous deployment

Toyota Kata Movement

2009 Mike Rother
Lean missed a key management practice "Improvement Kata"
→ process for continuous improvement @ organization level

Lean Startup Movement

2011 Eric Ries
- cont deployment techniques
- Min Viable product
- build-measure-learn
- CD technical patterns

Lean UX Movement

- gain confidence in business hypotheses before investing in features

Rugged Computing Movement

2011 - futility of securing applications and environments late in the lifecycle

frame non-functional requirements
"the -ilities"

Lean Software Development

- Agile practice based on lean Manufacturing
- 2003 Mary + Tom Poppendieck
  - 7 principles
  - 22 tools
- The fundamental principle is "eliminate waste"

7 Principles:

The Fundamentals

7 Principles:

1. Eliminate (waste = anything that doesn't add value to customer)
   - partially done work
   - extra features
   - extra processes
   - task switching
   - defects
2. Build quality/integrity in
   "Quality is everyone's responsibility" W.E. Deming
   - verify quality is built into product and process

   e.g. refactorings
3. Amplify Learning / Create Knowledge
   - Development is constant learning
   - increase feedback
     → short increments to customer
4. Defer commitment
   - Make decisions at right time
   - Delay until we have more info

   e.g. Prius wasn't originally hybrid
5. Deliver fast
   - early/frequent feedback
   - allows for course correction
   - smaller batches enable faster delivery
6. Respect people
   - empower team
7. Optimize the whole
   - systems-level thinking

7 Wastes of lean

1. Work that is partially done
   → completed code not checked in
   → untested/undocumented code
   Delays value stream → Quality issues
2. Extra features
   → don't build more than the customer needs
3. Revisiting decisions
4. Handoffs
   → back and forth communication
5. Delays
6. Task switching
7. Defects

Defining DevOps and observe influence of Lean.

Roots:
- Lean principles/practices
- Deliver more value w/ less waste
- Strong focus on respect for people

Lean values:
- Breaking down silos
- Improve collaboration between Dev + Ops
- Cultural focus: Deliver value to customer

Key Acronyms:

CAMS

CALMS

Definition

"[illegible] is about humans.

CAMS

Culture
Automation
Lean
Measurement
Sharing

CALMS

"DevOps is about humans. DevOps is a set of practices and patterns that turn human capital into high-performance organizational capital." John Willis

---

Another definition (Gene Kim)

- collaborative working relationship between Dev and IT ops.
- results in a fast flow of planned work
- while simultaneously increasing reliability, stability, resilience, and security of production envt.

- DevSecOps ← security teams incorporated
- entire organization must transform.

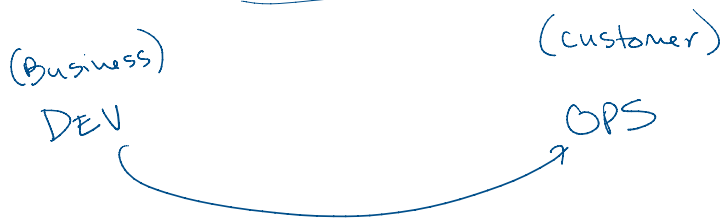
The Three Ways: principles from which all observed devops behaviors can be defined.

1. Systems thinking
2. Amplifying feedback loops
3. Culture of continuous experimentation + learning

#1 Systems thinking ≠ Siloed thinking

- emphasize perf. of entire system
- collaboration across functional lines

- focuses on IT-enabled Value Streams



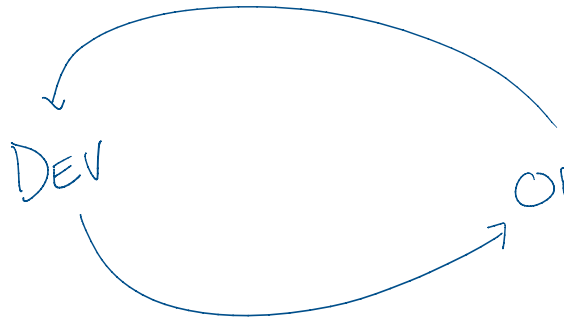
→ stop passing defects downstream
→ increase flow
→ profound understanding of system as a whole

Example : C. Kissler at Nordstrom

- Documenting value stream → saw defects being passed
- Data demonstrated sub-optimization
- led to automated testing, measuring quality

#2 Amplifying feedback loops

→ A process improvement initiative



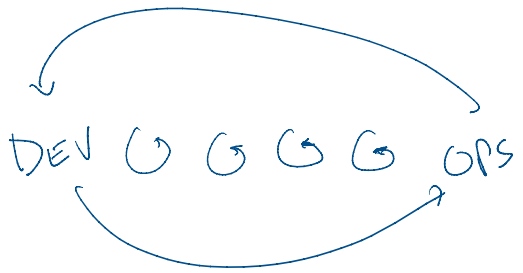
- Understand + respond to customers
- Shorten loop
- embed knowledge where it is needed
- build automated testing into pipeline
- embed ops engineers into dev teams

eg Starbucks

- Dashboard
- red when % test failed
- team could see issues and deliver feedback early + often

#3 Culture of Continual experimentation + learning

- encourage risk-taking and failing forward
- affirm that repetiton is a prereq to mastery



- Allocating time for improvement work
- Rewarding risk
- fault injection to increase resilience

- culture of innovation
- Requires strategic alignment w/ leadership

eg. Nike

- Innovation days
- Hackathons
- Test stores to experiment w/ new improvements
- provided capacity for learning