Anders Wallgren
Measuring DevOps - Key Metrics that Matter
We are here to talk metrics

The only valid measurement of code quality: WTFs/minute

Good code.

Bad code.

(c) 2008 Focus Shift

Electric Cloud
What we hear: Today’s software delivery challenges

Every business is a software business

- High cost (risk, time) per product release
- Manually operated, non-integrated tool chains
- Lack of shared visibility across Dev-QA-Ops
- No repeatability, predictability
- No traceability, auditability
- Inefficient infrastructure, low utilization
- Non-standard practices
DevOps by the numbers

<table>
<thead>
<tr>
<th></th>
<th>2015 (Super High vs. Low)</th>
<th>2014 (High vs. Low)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment Frequency</td>
<td>30x</td>
<td>30x</td>
</tr>
<tr>
<td>Deployment Lead Time</td>
<td>200x</td>
<td>200x</td>
</tr>
<tr>
<td>Mean Time to Recover</td>
<td>168x</td>
<td>48x</td>
</tr>
<tr>
<td>Change Success Rate</td>
<td>60x</td>
<td>3x</td>
</tr>
</tbody>
</table>

From: IT Revolution and Puppet Labs’ 2015 State of DevOps
DevOps by the numbers

High performers spend 22% less time on unplanned work and rework.

High-performing IT organizations report experiencing:
- 200x more frequent deployments
- 24x faster recovery from failures
- 3x lower change failure rate
- 2,555x shorter lead times

From: IT Revolution and Puppet Labs’ 2016 State of DevOps
Observation is fundamental to the scientific method

“You can't improve what you can't measure”
Why Metrics?

To make ourselves feel good?

To make ourselves feel bad?

To make pretty graphs?

“My number is bigger than your number”
Key principles of measuring software

Software’s most important quality is its adaptiveness and ease of change.

- **Efficiency**: Rate and cost per release
- **Effectiveness**: Ability to add more value

Best Practices

OBJECTIVITY

FOCUS ON OUTCOMES

IDENTIFY PREDICTORS

WATCH SIGNAL TO NOISE RATIO

Electric Cloud

atlanta GEORGIA
July 25-29, 2016
Best Practices

- **Avoid metrics that can be gamed**
- **Value team performance**
- **Shaming encourages gaming**
How

- Automatic
- Unobtrusive
- Not only production
- One pane of glass
- Measure-ability
- Technology
How

VANITY METRICS

Actionable
Accessible
Auditable
"Tickets closed"
"Customer Sat"
"Uptime" "MTTR"
Types of metrics

• Internal: inside-out measurements of efficiency
  -(cost/time/materials)
  -Tech progress, product pipeline trends
    and resource utilization
  -Measures of efficiency and resource consumption
• External: outside-in measurements of effectiveness
  -(value delivered)
  -Quality, usefulness, performance, and business outcomes
  -Effectiveness and value delivered
• Culture:
  -(objective and subjective trends in team dynamics)
  -Process overhead, trustworthiness, shared objectives, morale,
    motivation, team/product/enterprise identity
## What to measure in the pipeline

<table>
<thead>
<tr>
<th>DEV/CI</th>
<th>QA</th>
<th>Deploy</th>
<th>Release</th>
<th>Operate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development lead time</td>
<td>Idle time</td>
<td>Deployment lead time</td>
<td>Release frequency</td>
<td>MTTR</td>
</tr>
<tr>
<td>Rework required by defects, build breakage, downtime</td>
<td>Defects discovered/escaped, impact of defects</td>
<td>Deployment frequency, duration</td>
<td>Time/cost per release</td>
<td>Cost/frequency of outages</td>
</tr>
<tr>
<td>Idle time</td>
<td>MTTD</td>
<td>Change success rate</td>
<td>Predictability</td>
<td>On-call after business hours</td>
</tr>
<tr>
<td>Work-in-progress and technical debt</td>
<td>MTTR</td>
<td></td>
<td></td>
<td>Performance / utilization</td>
</tr>
<tr>
<td>Cycle time</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**-- Cycle Time --**

**-- Visibility --**

**-- Scale --**
What to measure in the pipeline

- Idle time
- Defects discovered/escaped, impact of defects
- MTTD
- Release frequency
- Predictability
- Deployment lead time
- Deployment frequency, duration
- Change success rate
- MTTR
- Cost/frequency of outages
- On-call after business hours
- Performance/utilization
- Development lead time
- Rework required by defects, build breakage, downtime
- Idle time
- Work-in-progress and technical debt
- Cycle time

-- Cycle Time --

-- Visibility --

-- Scale --
Dev/CI: Where to Start?

• Trunk-based development (or very-short-lived branches)
• Self-service automation for environment provisioning
• *-as-code
• Build quality in (less unplanned work downstream)
• Build security in (less unplanned work downstream)
What to measure in the pipeline

- Development lead time
- Rework required by defects, build breakage, downtime
- Idle time
- Work-in-progress and technical debt
- Cycle time
- Idle time
- Defects discovered/escaped, impact of defects
- MTTD

- Deployment lead time
- Deployment frequency
- MTTR
- Change rate
- MTTR

- Release frequency
- MTTR

- Operate

--- Cycle Time ---
--- Visibility ---
--- Scale ---
QA: Where to Start?

- Automated testing
- Fidelity of environments vs. prod
- Self-service automation for environment provisioning
- Continuous Delivery
What to measure in the pipeline

<table>
<thead>
<tr>
<th>DEV/CI</th>
<th>QA</th>
<th>Deploy</th>
<th>Release</th>
<th>Operate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development lead time</td>
<td>Idle time</td>
<td>Deployment lead time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rework required by defects,</td>
<td>Defects discovered/</td>
<td>Deployment frequency, duration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>build breakage, downtime</td>
<td>escaped, impact of</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>defects</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Idle time</td>
<td>MTTD</td>
<td>Change success rate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work-in-progress and</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>technical debt</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cycle time</td>
<td></td>
<td>MTTR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-- Cycle Time --

-- Visibility --

-- Scale --
Deploy: Where to Start?

- Automate all the things
- The first deployment shouldn’t be to PROD...or even PRE-PROD...
- Continuous delivery
  - Improves deployment frequency, reliability
- Artifact version control
- *-as-code
What to measure in the pipeline

- Development lead time
- Rework required by defects, build breakage, downtime
- Idle time
- Work-in-progress and technical debt
- Cycle time
- Visibility
- Scale

-- Cycle Time --

-- Visibility --

-- Scale --

- Operate
- Release
- Deployment lead time
- Deployment frequency, duration
- Change success rate
- MTTR
- Cost/frequency of outages
- On-call after business hours
- Performance / utilization

- DEV/CI
- QA
- Deploy
- Release frequency
- Time/cost per release
- Predictability
Release: Where to Start?

- Model the software delivery pipeline
  - Ensures reuse, predictability, visibility
- Fidelity of everything -- tools, processes, environments
- Visibility
What to measure in the pipeline

DEV/CI
- Development lead time
- Idle time
- Work-in-progress and technical debt
- Cycle time

QA
- Idle time
- Defects discovered/escaped, impact of defects
- MTTD

Deploy
- Deployment lead time
- Deployment frequency, duration
- Change success rate
- MTTR

Release
- MTTA
- Time/cost per release
- Predictability

Operate
- MTTR
- Cost/frequency of outages
- On-call after business hours
- Performance/utilization

-- Cycle Time--

-- Visibility--

-- Scale--
Operate: Where to Start?

• Version control all artifacts (rollback, governance, visibility)
• Monitor health of systems and applications
• Self-service provisioning of environments
• Shared infrastructure to drive down opex/capex
What to measure in the pipeline

Dev/CI  QA  Deploy  Release  Operate

Cycle Time

-- Visibility --

-- Scale --
HUAWEI

140+ Countries

28 Joint Innovation Centers

14 Regional HQs

70K R&D Employees

45 Training Centers

150K Employees Worldwide

Revenue by Geo

- 14.5% Americas
- 33.4% China
- 35.1% EMEA
- 17% Asia Pacific

(2012)
10K+ Releases/year
1M+ System Integrations/year
100K Builds/day
30M Lines of Code
100M Test cases run/day
480K Code reviews/year
100+ Applications
2 min For Test System Provisioning
$2.5B Online sales, 2014
31->8 People per release (before->after)
## Gap and Huawei: Two tales of DevOps at scale

<table>
<thead>
<tr>
<th></th>
<th>Developer Build</th>
<th>Production Build</th>
<th>Regression Test</th>
<th>Full Test</th>
<th>Feature Delivery Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Huawei - Before</strong></td>
<td>10 minutes</td>
<td>300 minutes</td>
<td>240 minutes</td>
<td>24 hours</td>
<td>30 days</td>
</tr>
<tr>
<td><strong>Huawei - After</strong></td>
<td>1 minute</td>
<td>10 minutes</td>
<td>60 minutes</td>
<td>6 hours</td>
<td>7 days</td>
</tr>
<tr>
<td><strong>Gap - Before</strong></td>
<td>20 minutes</td>
<td>150 minutes</td>
<td>300 minutes</td>
<td>24 hours</td>
<td>15 days</td>
</tr>
<tr>
<td><strong>Gap - After</strong></td>
<td>20 minutes</td>
<td>120 minutes</td>
<td>150 minutes</td>
<td>6 hours</td>
<td>1 day</td>
</tr>
</tbody>
</table>

![Diagram](image-url)
What to measure in the pipeline

Visibility

-- Cycle Time --

-- Scale --

Dev/CI  QA  Deploy  Release  Operate

GE Healthcare

HUAWEI

HP

LOCKHEED MARTIN

QUALCOMM

URBAN SCIENCE.
What to measure in the pipeline

- Dev/CI
- QA
- Deploy
- Release
- Operate

- Cycle Time
- Visibility

Scale

Doing it like a unicorn!
Business KPIs
What to measure in the business

- Lead Time
- Stories Delivered
- Customer Satisfaction
- Acquisition, Retention Cost
Culture KPIs
What to measure in the culture

- Satisfaction
- Retention
# How to create a generative culture

<table>
<thead>
<tr>
<th>Pathological</th>
<th>Bureaucratic</th>
<th>Generative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low cooperation</td>
<td>Modest cooperation</td>
<td>High cooperation</td>
</tr>
<tr>
<td>Messengers shot</td>
<td>Messengers neglected</td>
<td>Messengers trained</td>
</tr>
<tr>
<td>Responsibilities shirked</td>
<td>Narrow responsibilities</td>
<td>Risk are shared</td>
</tr>
<tr>
<td>Bridging discouraged</td>
<td>Bridging tolerated</td>
<td>Bridging encouraged</td>
</tr>
<tr>
<td>Failure -&gt; scapegoating</td>
<td>Failure -&gt; justice</td>
<td>Failure -&gt; inquiry</td>
</tr>
<tr>
<td>Novelty crushed</td>
<td>Novelty -&gt; problems</td>
<td>Novelty implements</td>
</tr>
</tbody>
</table>

A typology of organizational cultures – R. Westrum
Top Seven Measures of Culture

1. Organizational investment in DevOps
2. The experience and effectiveness of team leaders
3. Continuous delivery practices
4. Achieving “win-win” outcomes for dev, ops, and infosec teams
5. Organizational performance
6. Deployment pain
7. Lean management practices

From: IT Revolution and Puppet Labs’ 2015 State of DevOps
Predictors of Strong Performance

1. Peer-reviewed change approval process
2. Version control for all production artifacts
3. Proactive monitoring
4. High-trust organizational culture
5. Win-win relationship between dev and ops
Transformative Benefits

FASTER DEVELOP TO DEPLOY

- FamilySearch: 10 min, 90 days, 99% improvement
- E*TRADE: 10 min, 120+ min, 12X improvement
- GAP: 6 hours, 24 hours, 75% improvement
- Lockheed Martin: minutes, 20 days, 90% improvement

FASTER AUDITABILITY

- Who, what, when, how
- 90% improvement
Resources

- http://devops.com/2015/01/26/metrics-devops/
- https://blog.appdynamics.com/devops/quantified-devops/
- http://www.slideshare.net/jedi4ever/devops-metrics
Questions?
Thank You!

Anders Wallgren  |  @anders_wallgren
Measuring DevOps - Key Metrics that Matter