Outline

1. Introduction to Continuous Delivery
2. Continuous Delivery Benefits
3. Continuous Delivery Requirements
4. Continuous Deployment

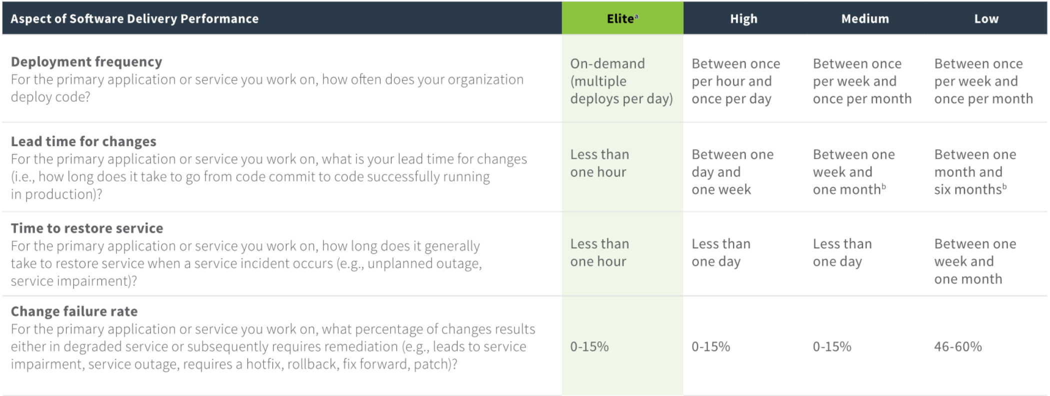
A Definition

> “*Continuous Delivery is a software development discipline where you build software in such a way that the software can be released to production at any time.*”
>
> — Martin Fowler

Benefits of CD

- Faster reaction times
- Reduced risk; increased stability
- Exposed inefficiencies and costs $\implies$ increased productivity
- Flexible release options

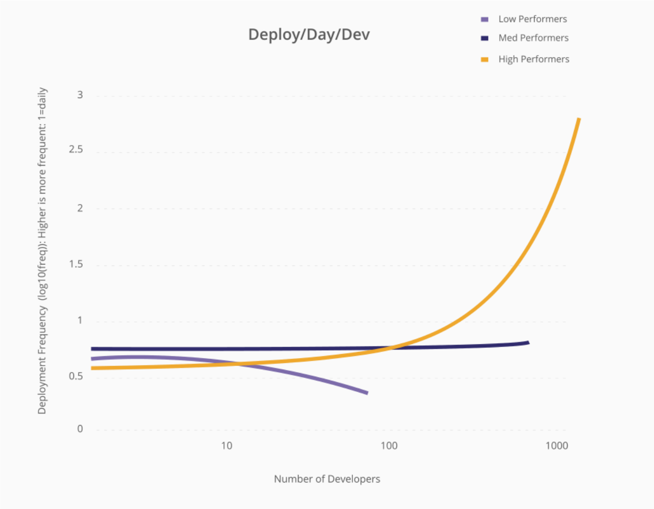
Software Delivery Performance



Aspect of Software Delivery Performance	Elite[a]	High	Medium	Low
Deployment frequency For the primary application or service you work on, how often does your organization deploy code?	On-demand (multiple deploys per day)	Between once per hour and once per day	Between once per week and once per month	Between once per week and once per month
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code commit to code successfully running in production)?	Less than one hour	Between one day and one week	Between one week and one month[b]	Between one month and six months[b]
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident occurs (e.g., unplanned outage, service impairment)?	Less than one hour	Less than one day	Less than one day	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes results either in degraded service or subsequently requires remediation (e.g., leads to service impairment, service outage, requires a hotfix, rollback, fix forward, patch)?	0-15%	0-15%	0-15%	46-60%

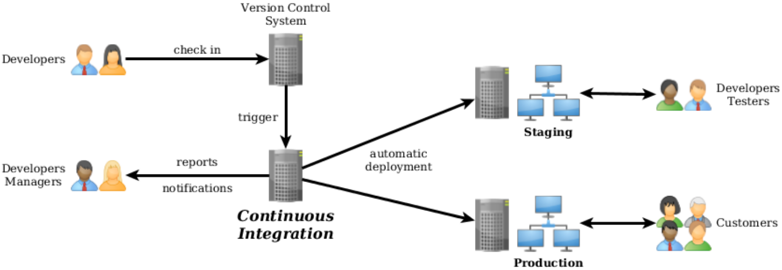
Source: Accelerate: State of DevOps 2018 by DORA
https://cloudplatformonline.com/2018-state-of-devops.html

Scaling Development Teams — Brooks's Law



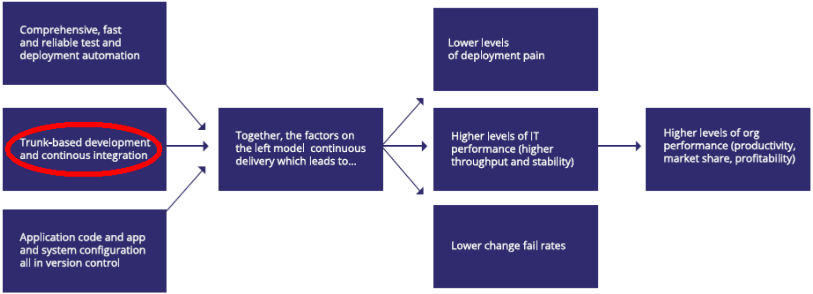
Source: 2015 State of Dev Ops Report by puppet labs
https://puppet.com/resources/white-paper/2015-state-of-devops-report

Infrastructure



Source: U Meding
http://uwemeding.com/making-the-case-for-continuous-integration/

Continuous Delivery Factors/Ingredients and Effects



Comprehensive, fast and reliable test and deployment automation

Trunk-based development and continous integration

Application code and app and system configuration all in version control

Together, the factors on the left model continuous delivery which leads to...

Lower levels of deployment pain

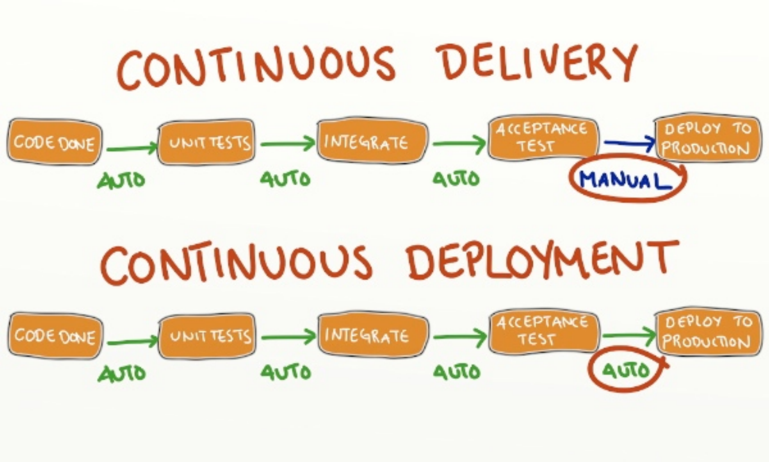
Higher levels of IT performance (higher throughput and stability)

Lower change fail rates

Higher levels of org performance (productivity, market share, profitability)

Source: 2015 State of Dev Ops Report by puppet labs
https://puppet.com/resources/white-paper/2015-state-of-devops-report

Continuous Delivery versus Continuous Deployment



Source: Y Sundman
http://blog.crisp.se/2013/02/05/yassalsundman/continuous-delivery-vs-continuous-deployment

Questions

- Why should we even try to achieve Continuous Delivery?
- What about software quality - how will features be tested if we have to ship at any point?
- How will we deal with the fact that at any given point in time some features are not complete?
- How do we get a new feature coded on a developer's laptop into the hands of a user?
- How often should the team collaborate and integrate their work?