Introduction to DevOps

20. August 2015

Leif Sørensen, CD Coach, partner and co-founder at Praqma
les@praqma.net

Praqma

- Continuous Delivery & DevOps experts and evangelists
- Tools & Automation experts
- We help customers with practical implementation of their development process
- 7 years, 20 employees, offices in Copenhagen, Aarhus, Oslo & Stockholm
- New venture: Container Solutions Denmark, together with Trifork
- Jenkins CI User Events: Copenhagen, 2011 - 2015
- CoDe Conferences: Oslo, Stockholm, Copenhagen, 2014,15
- Next:Continuous Delivery & DevOps Conference, Copenhagen, October 7th
- Service partner to:

CloudBees

IBM Rational software

Atlassian

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

How to get started

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

How to get started

Traditional Dev/Ops organisation

Development

- Responsible and measured on delivery of business functionality
- Require and like changes
- Poor understanding of writing “operational” applications

Operations

- Responsible and measured on stability & availability
- Dislike changes
- Maybe even outsourced

Traditional Dev/Ops organisation

- No common tools
- Handover points, check points (operations desperately trying to control ensure deliveries are living up to their demands)
  - Don’t catch problems
  - Slow and expensive
- Support is difficult, and often across Dev & Ops
- A lot of ‘blame games’
- Development projects and support are also separated - making it worse

Operations handover

Operations handover

Operations handover

Operations handover

Traditional Dev/Ops organisation

2 organisations with very different priorities and goals, different tools, different attitude and mentality, often 2 companies, working together to implement a pipeline for delivering software from developer to production

Bound to fail

Andon cord

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

How to get started

What is DevOps?

Is it a culture?

Is it a job title?

Is it a team?

Is it a way of organizing?

Is it a tool stack?

Or just a way of thinking?

Definition of DevOps

DevOps is agile IT operations delivery, required to match the cadence of agile IT development.

DevOps is the philosophy of unifying Development and Operations at the culture, practice, and tool levels, to achieve accelerated and more frequent deployment of changes to Production (The IT Sceptic)

DevOps emerged in 2009 when a group of Belgian developers hosted DevOps Days, which advocated collaboration between developers and operational staff

Definition of DevOps

The term “DevOps” typically refers to the emerging professional movement that advocates a collaborative working relationship between Development and IT Operations, resulting in the fast flow of planned work (i. e., high deploy rates), while simultaneously increasing the reliability, stability, resilience and security of the production environment (Gene Kim).

Operations tasks

- Architectural layout
- Performance
- Sizing
- Provisioning
- User management
- "Iron and wire”
- Quality gateways
- Deployment
- Database management
- Job scheduling
- Survelliance and monitoring
- Incident management
- 1st level support and dispatching

DevOps Deliveries

A dynamic platform, where resources can be provisioned automatic on the fly

The resources must be available for development, test and production

An automatic CD Pipeline, that goes the whole way to production

Quality Assurance

Continuous Delivery & DevOps provides new possibilities for test environments for manual and automated tests

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

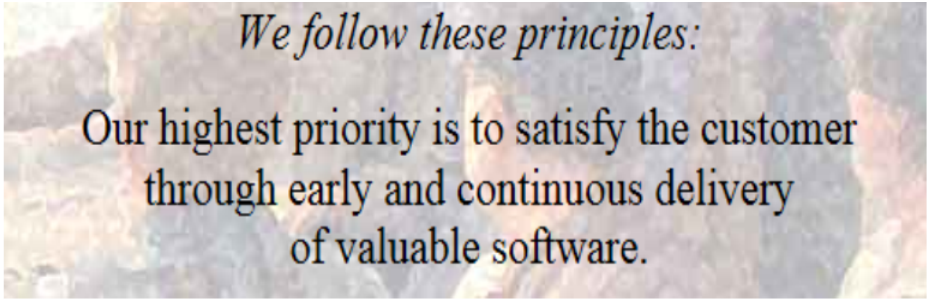
NoOps

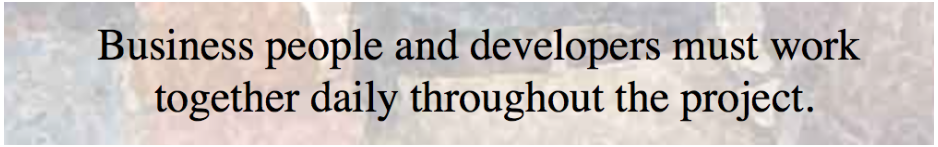
Challenges - Organisation /Maturity / Technical debt

How to get started

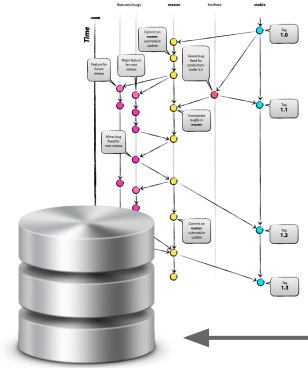
Agile Manifesto

- Four doctrines
- Twelve Principles



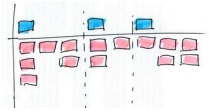


CD Story-line



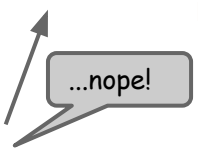
VCS

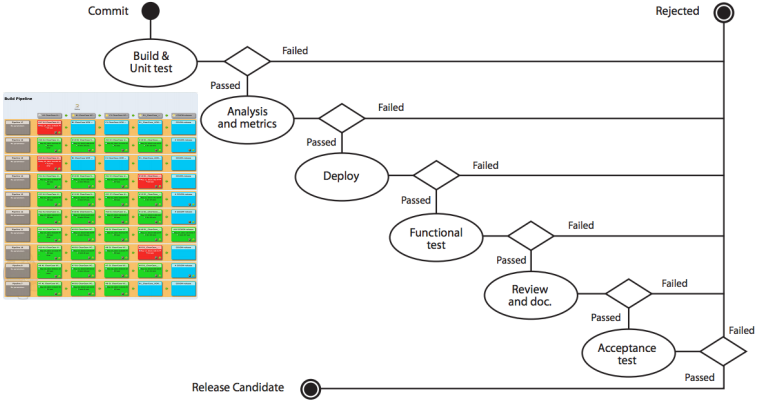
Developer



Work item

Automation Platform

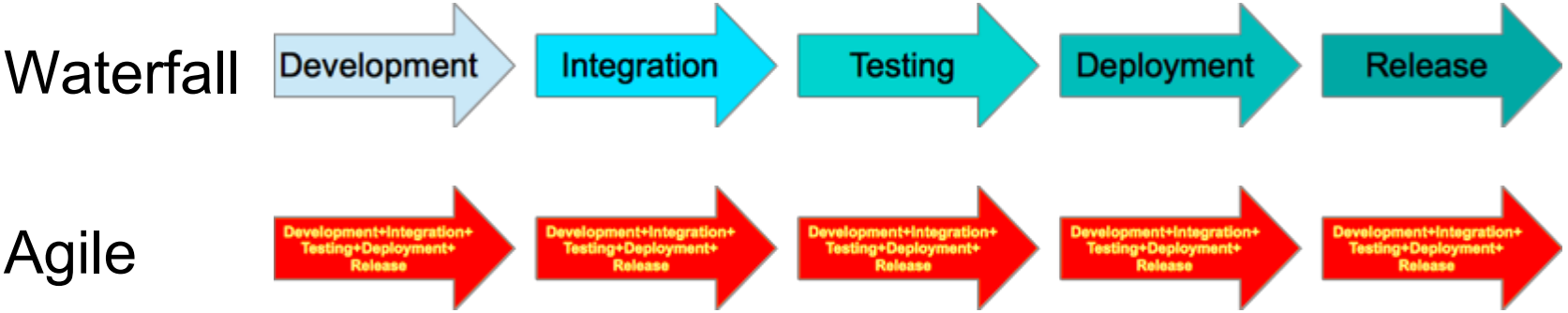




Continuous Delivery Pipeline

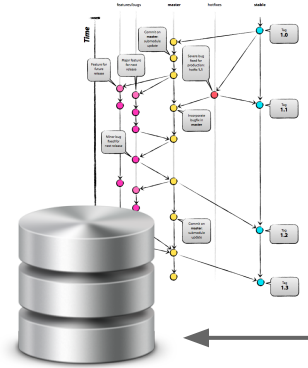
Done done

Agile defies phases



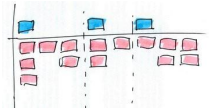
Watergile? Agilefall?

CD Story-line



VCS

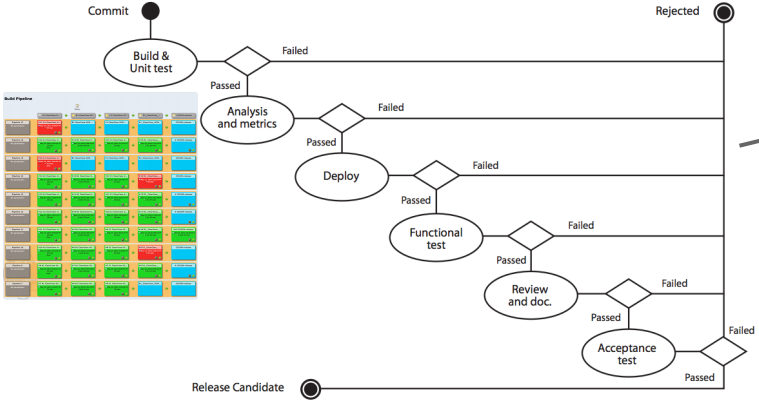
Developer



Work item

Automation Platform

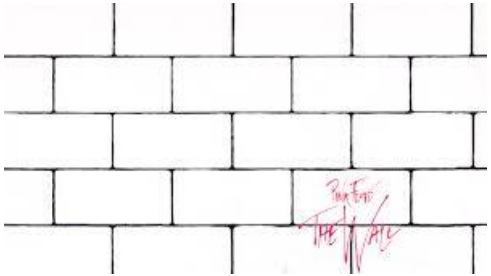
...nope!



Continuous Delivery Pipeline

Done done

Operations



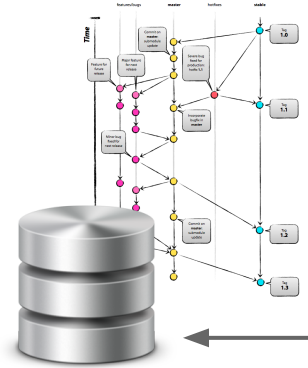
Is this a problem?

Unachieved goals:

- Unpredictable deliveries
- No visibility into the progress
- Cannot deliver fast
- Building up technical debt
- Quality is put on as an afterthought

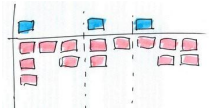
It is very expensive, to develop software that you do not put into production

CD Story-line



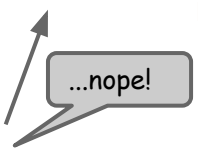
VCS

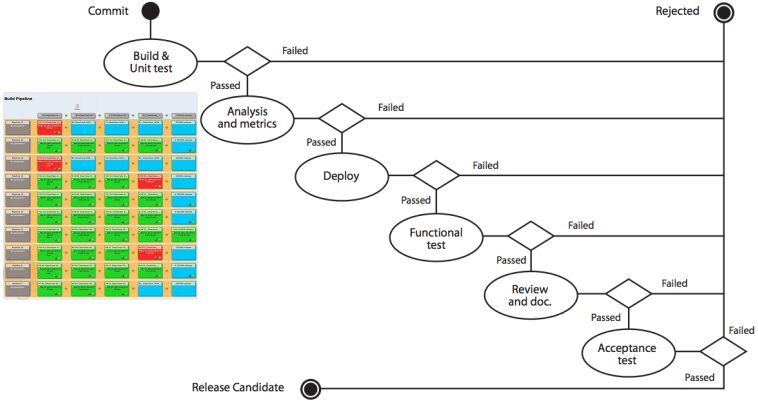
Developer



Work item

Automation Platform





Continuous Delivery Pipeline

Done done

Continuous Delivery

- When we implement CD, we do a lot of operations work - that should be reused
- Highly automated process
- Requires a lot of environments for build, test,...
- Require ability to dynamically provision new environments
- Dev needs Ops for this
- CD can help creating trust between Dev & Operations
- You don’t “do” DevOps. You “do” Continuous Delivery

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

How to get started

Infrastructure & Tools

- Important to see the tool stack as one integrated tool stack for Development and Operations
- Infrastructure as code
- Configuration as code
- You need tools for CI Server, Configuration management, Provisioning,...

Infrastructure on demand

You can buy infrastructure a lot cheaper and simpler than making a big IT Operations outsourcing deal with XXX

Tools


it·forum
PRAQMA


Windows Azure


amazon
web services™


vmware®


Atlassian
Bamboo


Jenkins


ANSIBLE


uDeploy


XL DEPLOY


puppet
labs


CHEF


docker

Tools - Docker

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

How to get started

The forerunners - Unicorns

Amazon, Facebook, Twitter, Google, Netflix

Internet startups, the application is the business, more focus on new functionality than errors (?), can test in production (?),...

NoOps

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

How to get started

NoOps

Developers: we don’t need operations!

They are just a pain!

With the right tools and services - we would rather do it all ourselves!

Operations tasks

- Architectural layout
- Performance
- ~~Sizing~~
- ~~Provisioning~~
- User management
- "Iron and wire”
- ~~Quality gateways~~
- ~~Deployment~~
- ~~Database management~~
- ~~Job scheduling~~
- Surveillance and monitoring
- ~~Incident management~~
- 1st level support and dispatching

NoOps

- NoOps means developers can code and let a service deploy, manage and scale their code
- NoOps means automated systems managing app lifecycles, not SysAdms. “the point isn’t that ops are going away, but they’re going away for developers” (Derrick Harris at GigaOm)

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

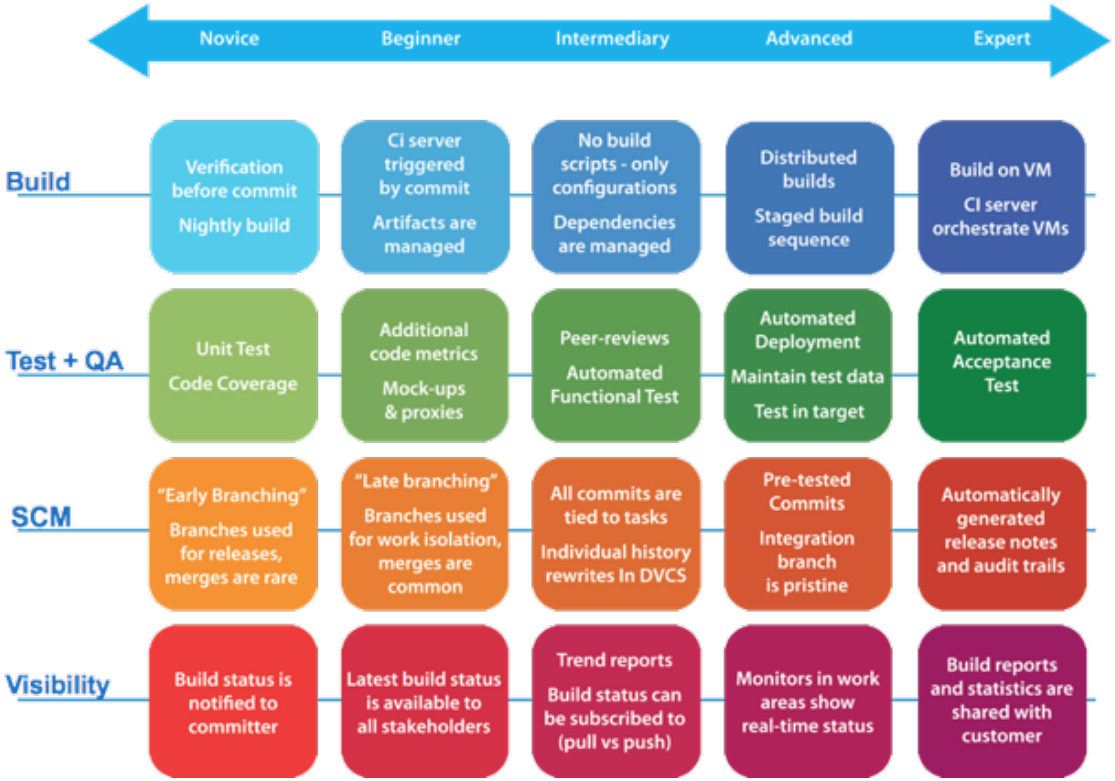
How to get started

Challenges - Organisation

- Your Dev and Ops people have conflicting goals
- Your Operations people have been beaten into defensive mode
- Everybody are just so used to the Dev versus Ops way of working

Challenges - Technical

Continuous Delivery Maturity



Operational Maturity

- Application versus database changes
- Non-monolithic architecture - Microservices
- Operational, testable, supportable applications
- Tracebility / Visibility
- Configuration, Infrastructure as code

Technical debt

Henrik Kniberg: The Solution to Technical Debt http://blog.crisp.se/2013/07/12/henrikkniberg/the-solution-to-technical-debt

Technical debt

Anything about your code & development environment that slows you down. For example:

- Unclear, unreadable code
- Lack of test automation, build automation, deployment automation, and anything else that could be automated that you do manually today
- Duplicate code
- Tangled architecture & unnecessarily complex dependencies
- Slow, ineffective tools
- Uncommitted code & long-lived branches (hides problems that will slow you down later)
- Important technical documentation that is missing or out-of-date
- Unnecessary technical documentation that is being maintained and kept up-to-date
- Lack of test environments
- Long build-test cycle & lack of continuous integration

Agenda

Traditional Development & Operations

Definition of DevOps

Why DevOps - Agile development & Continuous Delivery

Why DevOps - Infrastructure & Tools

Why DevOps - Unicorns - the internet startups

NoOps

Challenges - Organisation /Maturity / Technical debt

How to get started

How to get started

- You don’t “do” DevOps. You “do” Continuous Delivery (except Cloud - Private or public. That you just do!)
- Insource your operations, move it into the cloud, or find an outsourcing partner who can deliver DevOps (not tools, but services)
- Ensure that everybody have the same goal: developing and delivering working software
- Ensure you have people who want to develop and improve
- Think deployment & operations from the beginning of the project - Involve Operations

More materials

CD Maturity Paper: http://www.praqma.com/papers/cdmaturity

Continuous Delivery & DevOps Conferences, Jenkins CI User Events: http://www.code-conf.com/

Top 11 Things You Need To Know About DevOps, Gene Kim (http://www2.netuitive.com/rs/netuitive/images/Top11ThingsToKnowAboutDevOps.pdf)

The Convergence of DevOps, John Willis: (http://itrevolution.com/the-convergence-of-devops/)

The end - thank you!