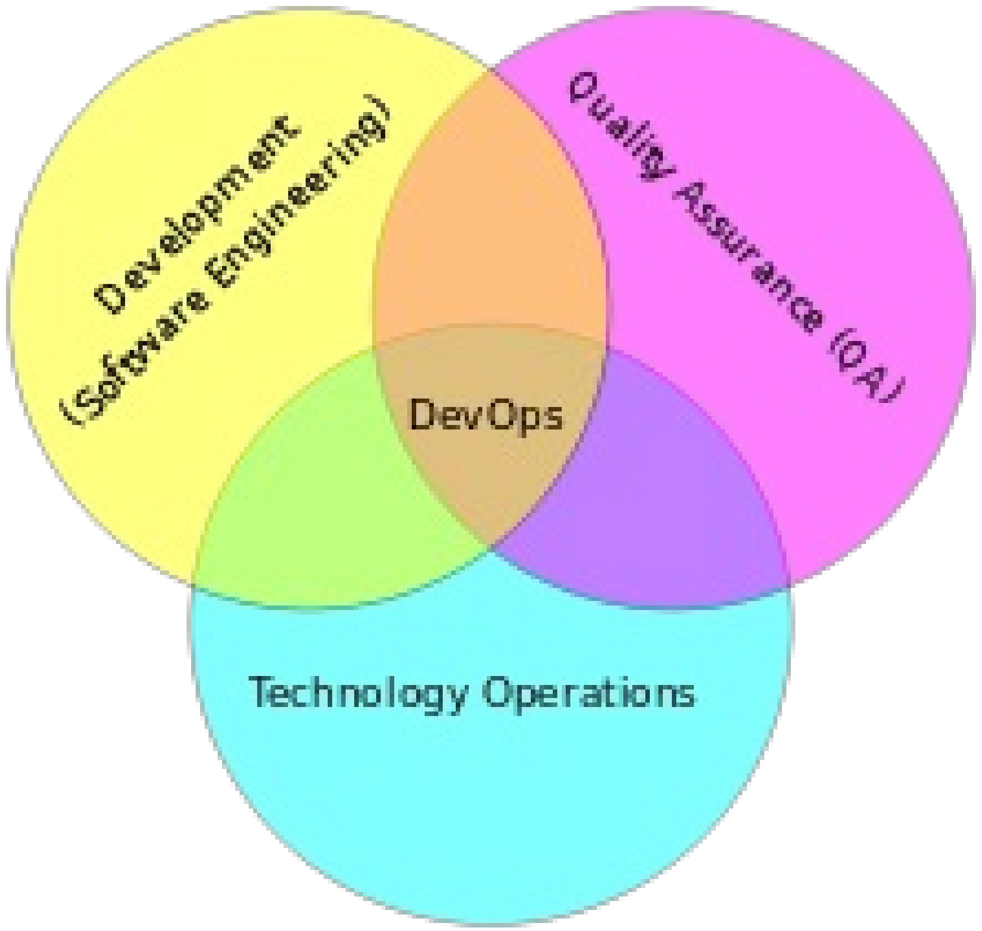
DevOps Best Practices, lessons learnt after burning fingers

Ankur Kumar

DevOps, what it is not?

- Software industry is obsessed by the jargons; OOP, OOAD, Cloud, now DevOps
- DevOps != just connecting Development and Operation together
- Not a shiny new term for old sysadmin mentality
- Not about continuing old and aged software development practices with wrapper of Agile added

So what the DevOps is?



- Its not a role but a culture and mindset
- Human factors are as important as the technologies
- Creating unified teams keeping Dev, QA and technical Operations together
- Everyone is involved in the entire architecture/design/development/testing/ operations decisions
- The whole Infrastructure is viewed As Code
- Nothing is manual and everything is automated as much as possible
- Its all about iterating in fast cycles by creating automated pipelines

But wait, we have seen this before!

- **W**rite **T**est **F**irst

  if you can't break it, you can't make it

- **K**eep **I**t **S**imple, **S**tupid

  most systems work best if they are kept simple rather than made complicated

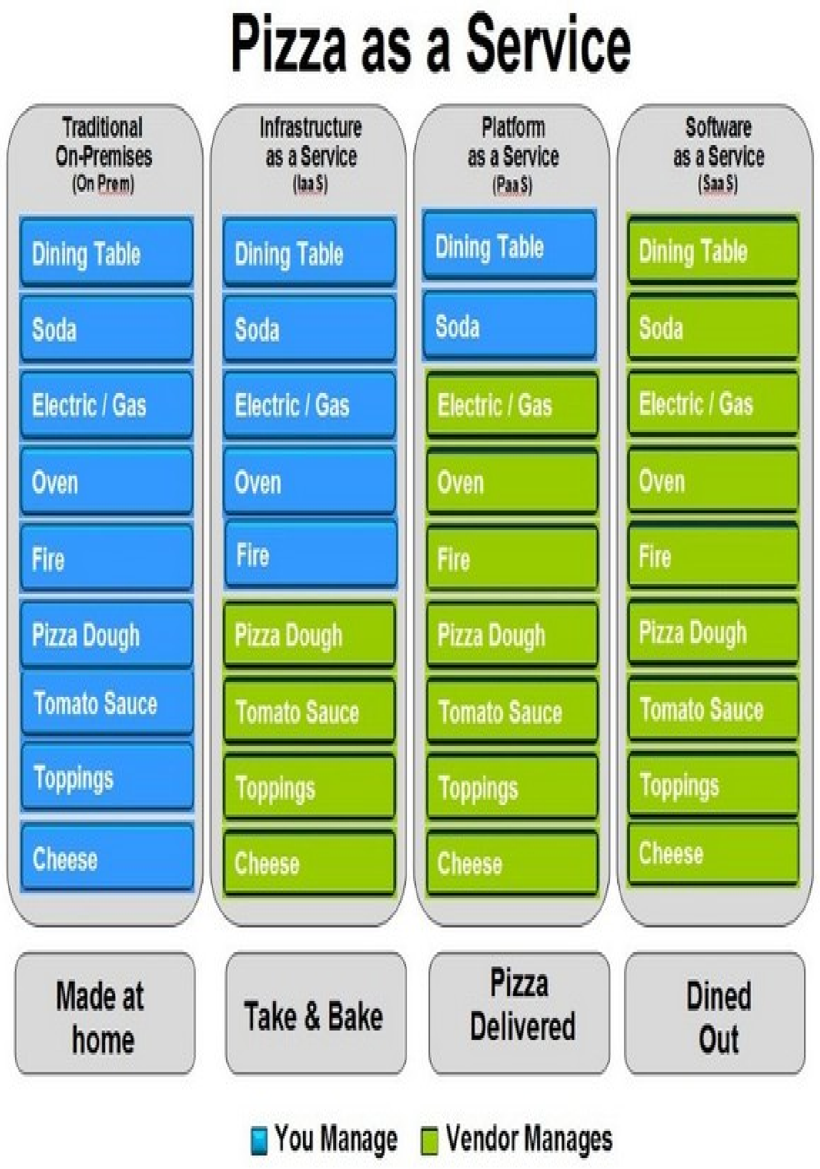
- **Y**ou **A**ren't **G**onna **N**eed **I**t

  do the simplest thing that could possibly work

- **D**on't **R**epeat **Y**ourself

  Every piece of knowledge must have a single, unambiguous, authoritative representation within a system

DevOps and Cloud



Pizza as a Service

Traditional On-Premises (On Prem)	Infrastructure as a Service (IaaS)	Platform as a Service (PaaS)	Software as a Service (SaaS)
🟦 Dining Table	🟦 Dining Table	🟦 Dining Table	🟩 Dining Table
🟦 Soda	🟦 Soda	🟦 Soda	🟩 Soda
🟦 Electric / Gas	🟦 Electric / Gas	🟩 Electric / Gas	🟩 Electric / Gas
🟦 Oven	🟦 Oven	🟩 Oven	🟩 Oven
🟦 Fire	🟦 Fire	🟩 Fire	🟩 Fire
🟦 Pizza Dough	🟩 Pizza Dough	🟩 Pizza Dough	🟩 Pizza Dough
🟦 Tomato Sauce	🟩 Tomato Sauce	🟩 Tomato Sauce	🟩 Tomato Sauce
🟦 Toppings	🟩 Toppings	🟩 Toppings	🟩 Toppings
🟦 Cheese	🟩 Cheese	🟩 Cheese	🟩 Cheese
Made at home	Take & Bake	Pizza Delivered	Dined Out

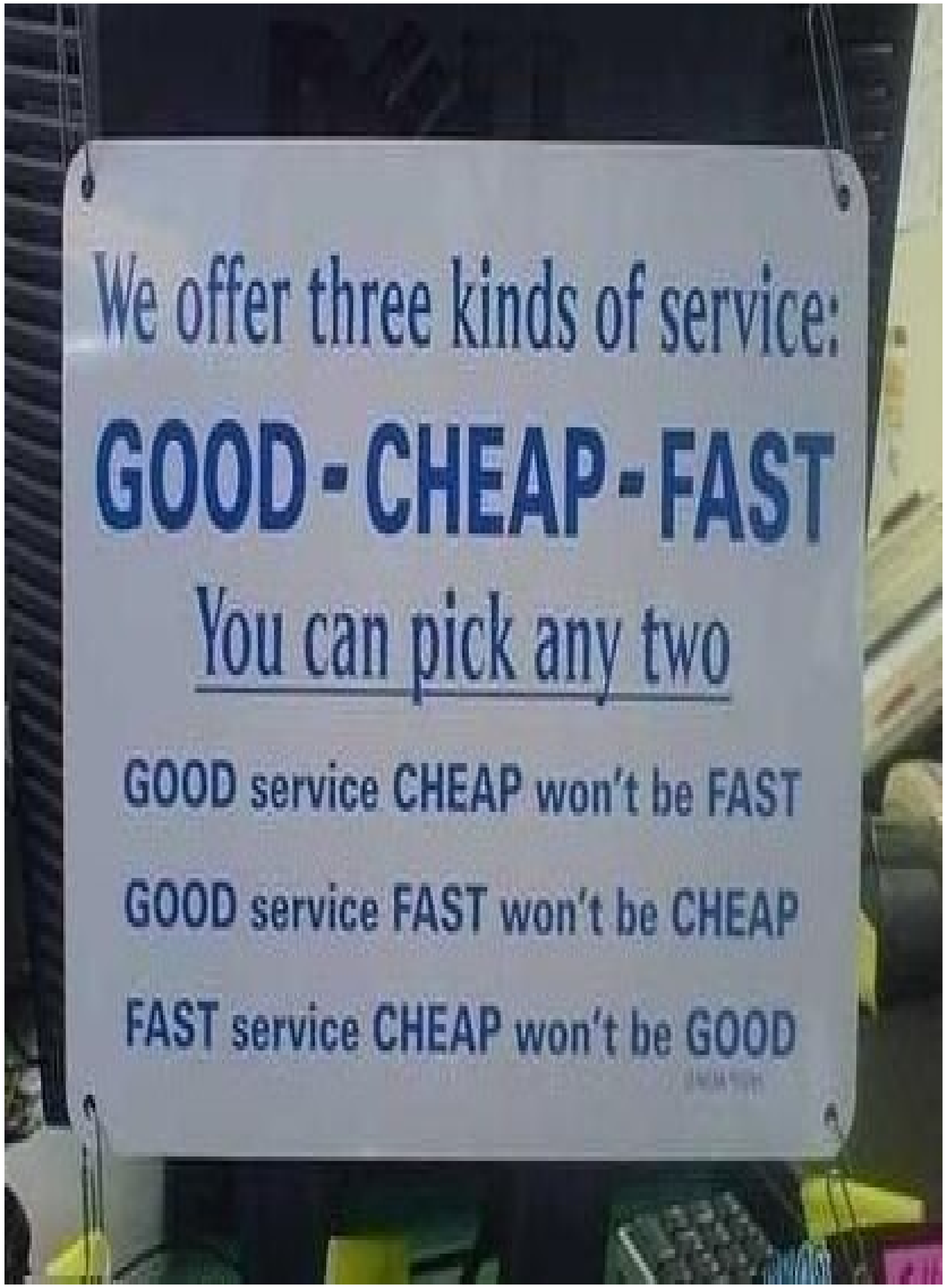
🟦 You Manage 🟩 Vendor Manages

- Both are about abstractions, elasticity and self service
- The secret sauce is the dynamism created by virtualization
- All the resources are software and/or services ready to get consumed
- Only the level of abstraction varies from highest to lowest i.e. IaaS, PaaS, SaaS

Fallacies of distributed computing

- The network is reliable.
- Latency is zero.
- Bandwidth is infinite.
- The network is secure.
- Topology doesn't change.
- There is one administrator.
- Transport cost is zero.
- The network is homogeneous.

CAP Theorem



We offer three kinds of service:

GOOD - CHEAP - FAST

You can pick any two

GOOD service CHEAP won't be FAST

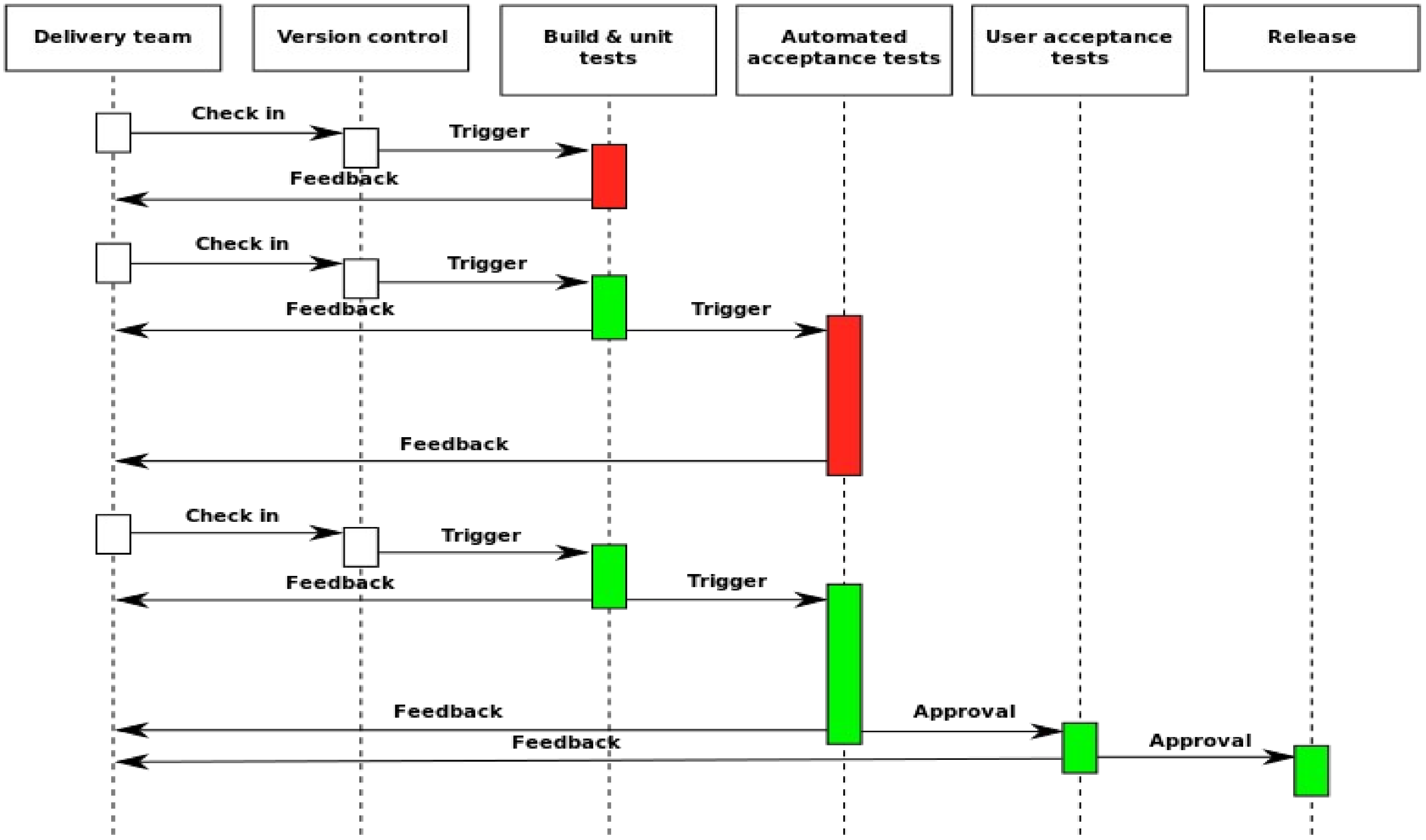
GOOD service FAST won't be CHEAP

FAST service CHEAP won't be GOOD

Any networked shared-data system can have at most two of three desirable properties:

- consistency (C) equivalent to having a single up-to-date copy of the data;
- high availability (A) of that data (for updates); and
- tolerance to network partitions (P).

Continuous Integration/Deployment/Delivery



Immutable Infrastructure

- The difference between the pre-virtualization model and the post-virtualization model can be thought of as the difference between pets and cattle
- One of the scariest things to ever encounter is a server that’s been running for ages which has seen multiple upgrades of system and application software
- Need to upgrade? No problem. Build a new, upgraded system and throw the old one away. New app revision? Same thing. Build a server (or image) with a new revision and throw away the old ones
- We write down, before the server is created, all the things we would like to be done on the server after it boots

Micro Services and Service Discovery

- Complex applications are composed of small, independent processes communicating with each other using language-agnostic APIs
- Services are small, highly decoupled and focus on doing a small task, facilitating a modular approach to system-building
- Service instances have dynamically assigned network locations and the set of service instances changes dynamically because of auto-scaling, failures, and upgrades
- Service discovery provides a highly available centralized registry to discover various services registered
- A required component for any containers based dynamic infrastructure

Tools of the trade

- More code more problem, less code less problem, no code no problem :-)
- Nothing/Minimum dependencies to install and operate
- Extensible for the custom functionalities
- Agnostic to various cloud providers
- Free Open Source, no/minimum community/enterprise kind of confusion

Ansible/Terraform

- Both describe infrastructure in abstracted modeling languages
- Ansible is a push mode cloud provisioning/configuration management and job orchestration tool
- The ansible CM provides idempotency for the official modules provided and uses ssh as the transport
- New modules could be coded in any of the programming languages
- Terraform is a very simple way to create cloud infrastructure in an idempotent way
- Works on dependency graphs and creates infrastructure pieces in parallel whenever possible
- Faster than the ansible as created in golang but the plugins could only be created in golang

Packer

- An open source tool for creating identical machine images for multiple platforms from a single source configuration
- Lightweight, runs on every major operating system, and is highly performant, creating machine images for multiple platforms in parallel
- Fully supports automated provisioning in order to install software onto the machines prior to turning them into images
- Historically, pre-baked images have been frowned upon because changing them has been so tedious and slow
- A must have for the images based immutable infrastructure

Consul/Docker

- Makes it simple for services to register themselves and to discover other services via a DNS or HTTP interface
- Pairing service discovery with health checking prevents routing requests to unhealthy hosts and enables services to easily provide circuit breaker
- Scales to multiple datacenters out of the box with no complicated configuration
- Flexible key/value store for dynamic configuration, feature flagging, coordination, leader election and more
- One of the few choices for containers service discovery

References

- https://en.wikipedia.org/wiki/DevOps
- https://en.wikipedia.org/wiki/Continuous_delivery
- http://radar.oreilly.com/2015/06/an-introduction-to-immutable-infrastructure.html
- https://blog.engineyard.com/2014/pets-vs-cattle
- http://chadfowler.com/blog/2013/06/23/immutable-deployments/
- https://www.nginx.com/blog/introduction-to-microservices/
- https://hashicorp.com/#products

Code/Demo/Q&A

Thanks!