SCALEOUT

Configuration Management Tools

Max Andersson

SCALEOUT

Cloud Computing

- What is Cloud Computing?
- Distribution of responsibilities
- Vendor Lock-in vs Vendor Agnostic

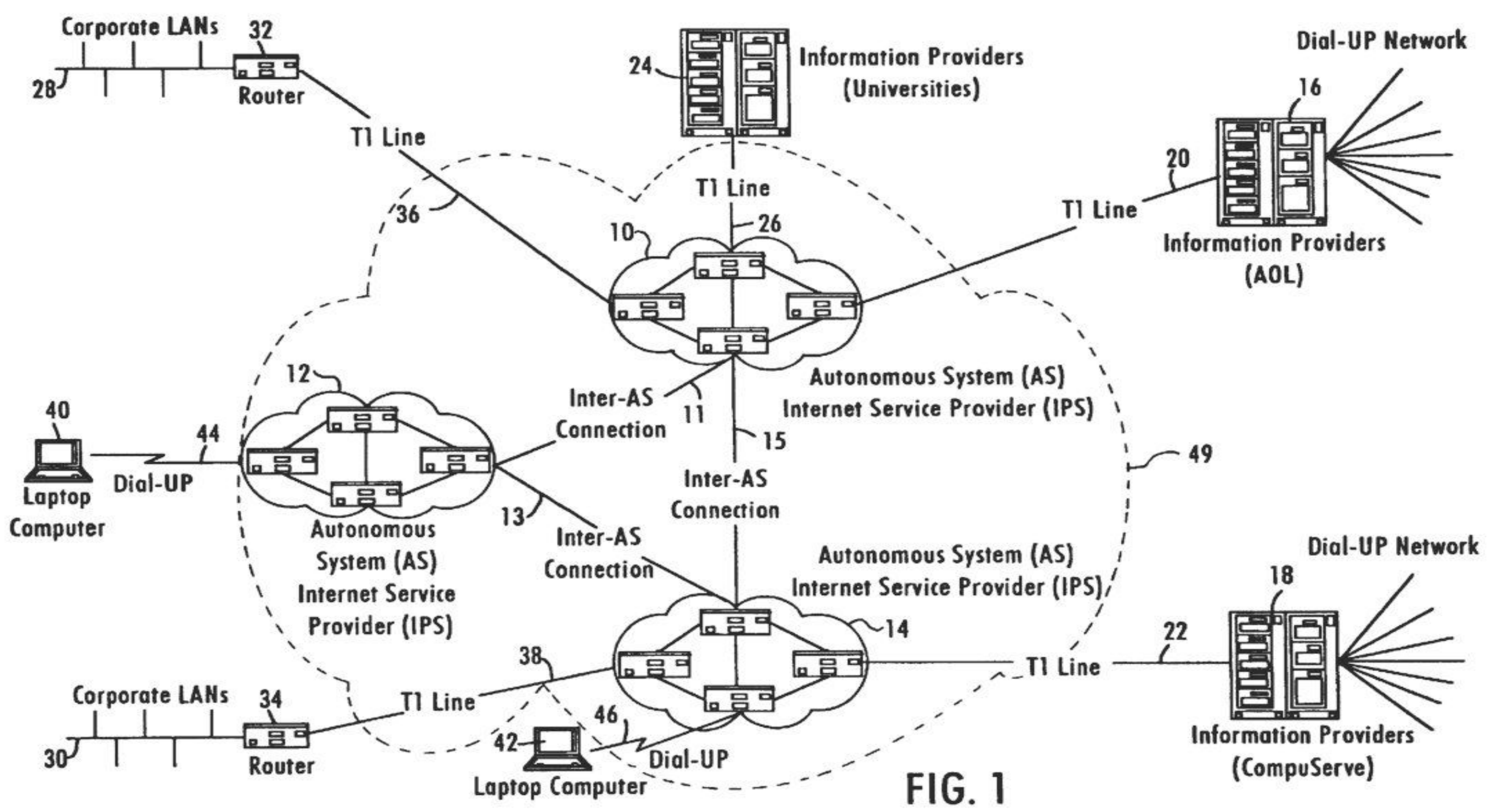

Corporate LANs
32
28
Router
T1 Line
24
Information Providers
(Universities)
Dial-UP Network
16
20
T1 Line
36
T1 Line
26
10
Information Providers
(AOL)
12
Autonomous System (AS)
Internet Service Provider (IPS)
40
Inter-AS
Connection
11
44
15
49
Laptop
Computer
Dial-UP
Inter-AS
Connection
13
Autonomous
System (AS)
Internet Service
Provider (IPS)
Inter-AS
Connection
Autonomous System (AS)
Internet Service Provider (IPS)
Dial-UP Network
18
14
22
38
T1 Line
Corporate LANs
34
T1 Line
30
Router
46
42
Dial-UP
Laptop Computer
Information Providers
(CompuServe)

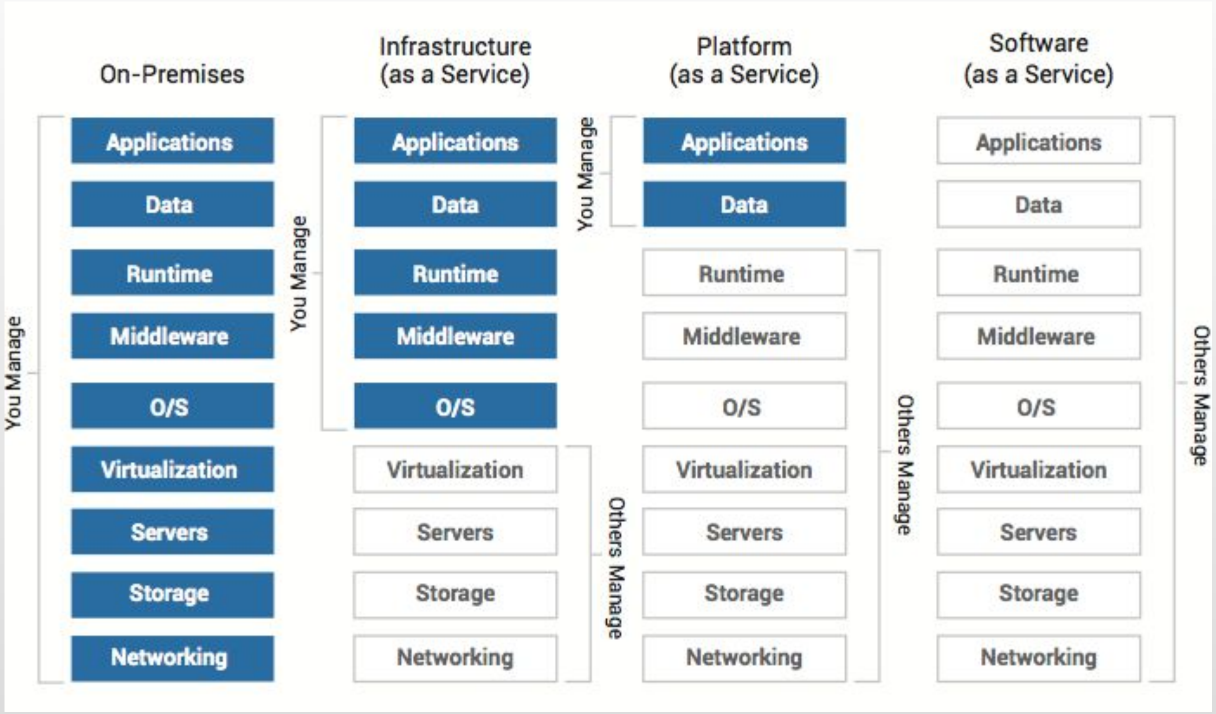
FIG. 1

Cloud Computing

- Essential Characteristics:
  - On-demand self-service
  - Broad network access
  - Resource pooling
  - Rapid elasticity
  - Measured service
- Service Models
  - Software as a Service (SaaS)
  - Platform as a Service (PaaS)
  - Infrastructure as a Service (IaaS)
  - CaaS, FaaS, MaaS, DPasS, MBaaS, etc... (derivatives)
- Deployment Models:
  - Private cloud
  - Community cloud
  - Public cloud
  - Hybrid cloud

https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

Cloud Computing



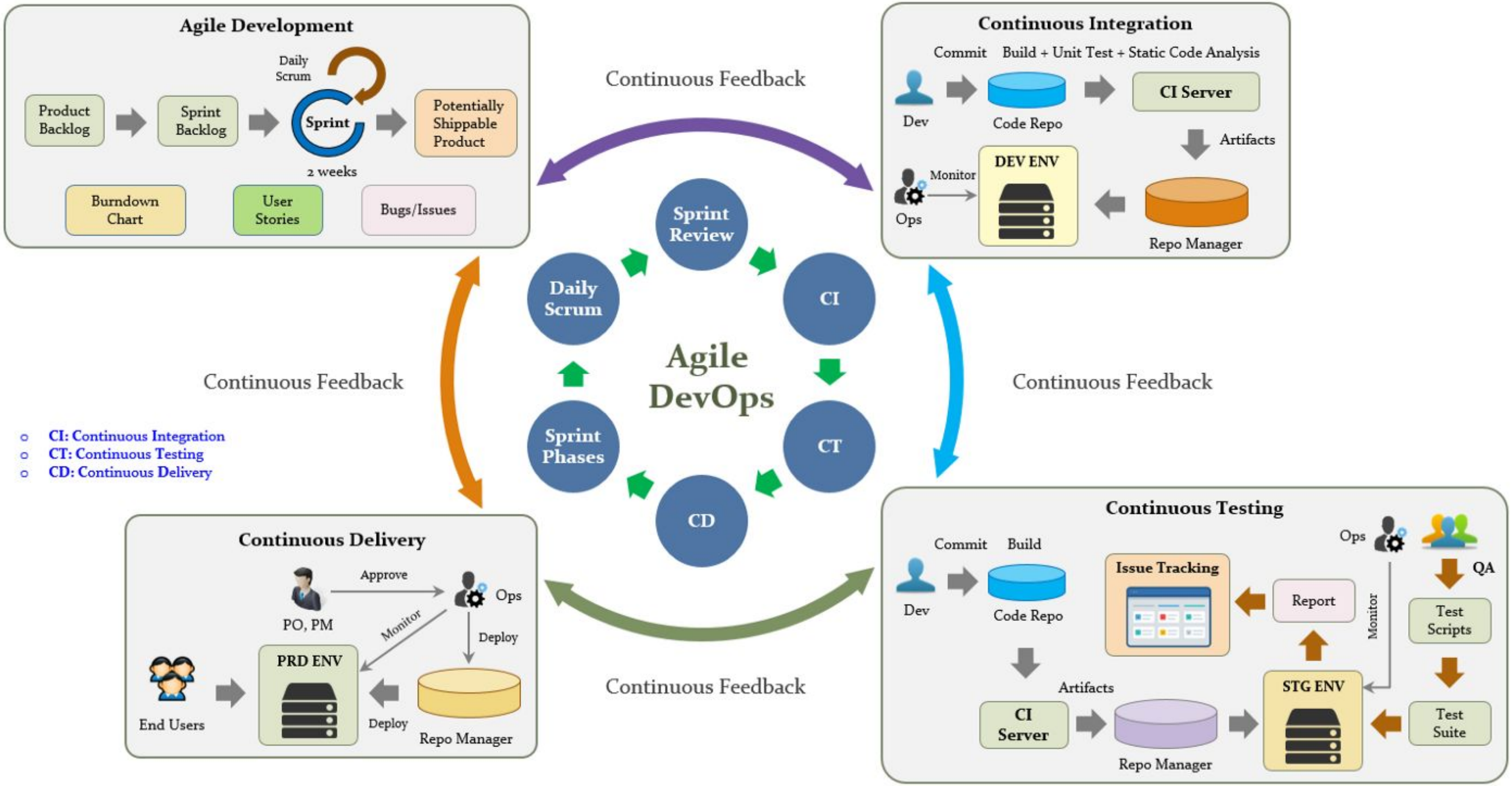
https://www.slideshare.net/RohitJnagal/docker-internals/4

SCALEOUT

DevOps - Intro

- What is DevOps?
- Definitions
- What problem does DevOps address?
- How does DevOps relate to agile?

SCALEOUT



- CI: Continuous Integration
- CT: Continuous Testing
- CD: Continuous Delivery

DevOps - The Problem

- Everything is software today
- Software need a server to run as a service
- Release Cycles are generally faster nowdays
- Making releases might be a business decision
- Managing a large number of servers or services.

- Disjoint groups of development and operations
- Operations are resistant to change
- Development is agile, Operations is usually static.
- Delays in getting to production is costly

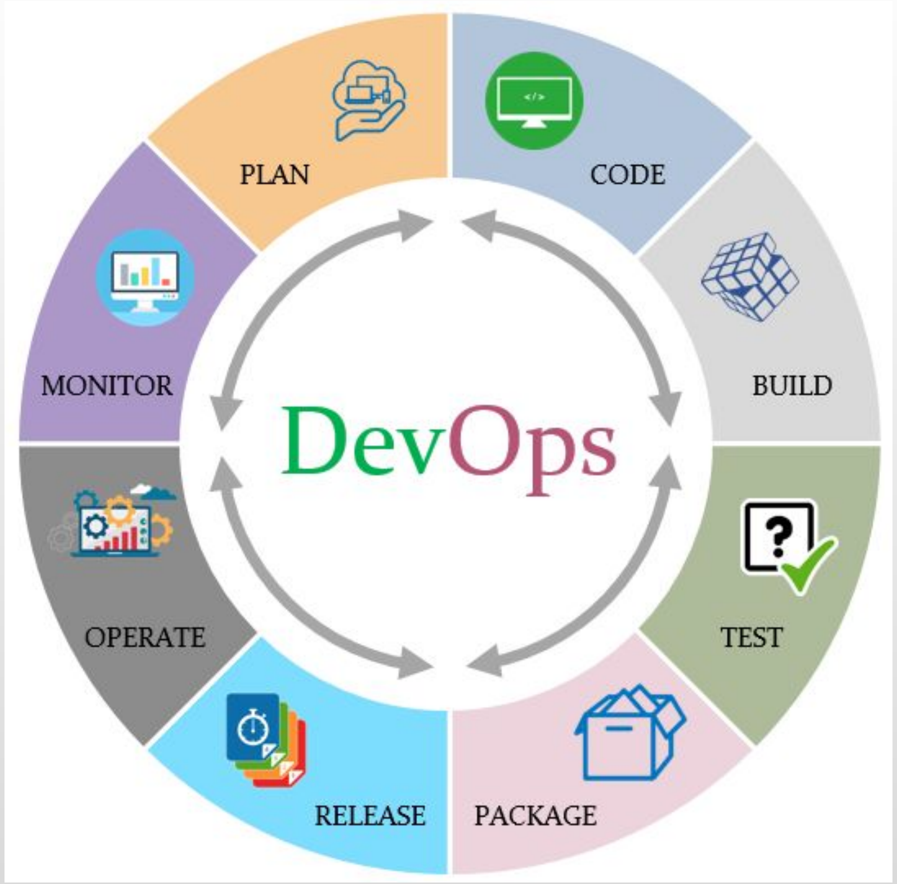
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

SCALEOUT

https://medium.com/@raycad.seedotech/devops-methodology-and-process-dde388eb65bd

DevOps - The Process

1. **Coding** – code development and review, source code management tools, code merging
2. **Building** – continuous integration tools, build status
3. **Testing** – continuous testing tools that provide feedback on business risks
4. **Packaging** – artifact repository, application pre-deployment staging
5. **Releasing** – change management, release approvals, release automation
6. **Configuring** – infrastructure configuration and management, infrastructure as code tools
7. **Monitoring** – applications performance monitoring, end-user experience



SCALEOUT

DevOps - Intro

- The Components
  - Volumes
  - Storage
  - Containers
    - CRI
    - Lifecycle
    - Stateful vs Stateless Applications
  - VM’s
  - Images
  - Snapshots
  - Networks/IP’s

Devops - The Pillars

- Infrastructure Automation
  - Infrastructure As Code
  - Application Deployment
  - Runtime Orchestration
- Continuous Delivery
- Reliability Engineering

SCALEOUT

Devops - Infrastructure Automation

- So what do we automate ?
    - Builds
    - Deployments
    - Testings
    - Monitoring
    - Self-Healing
    - System Rollouts
    - System Configuration

SCALEOUT

Devops - Infrastructure as Code

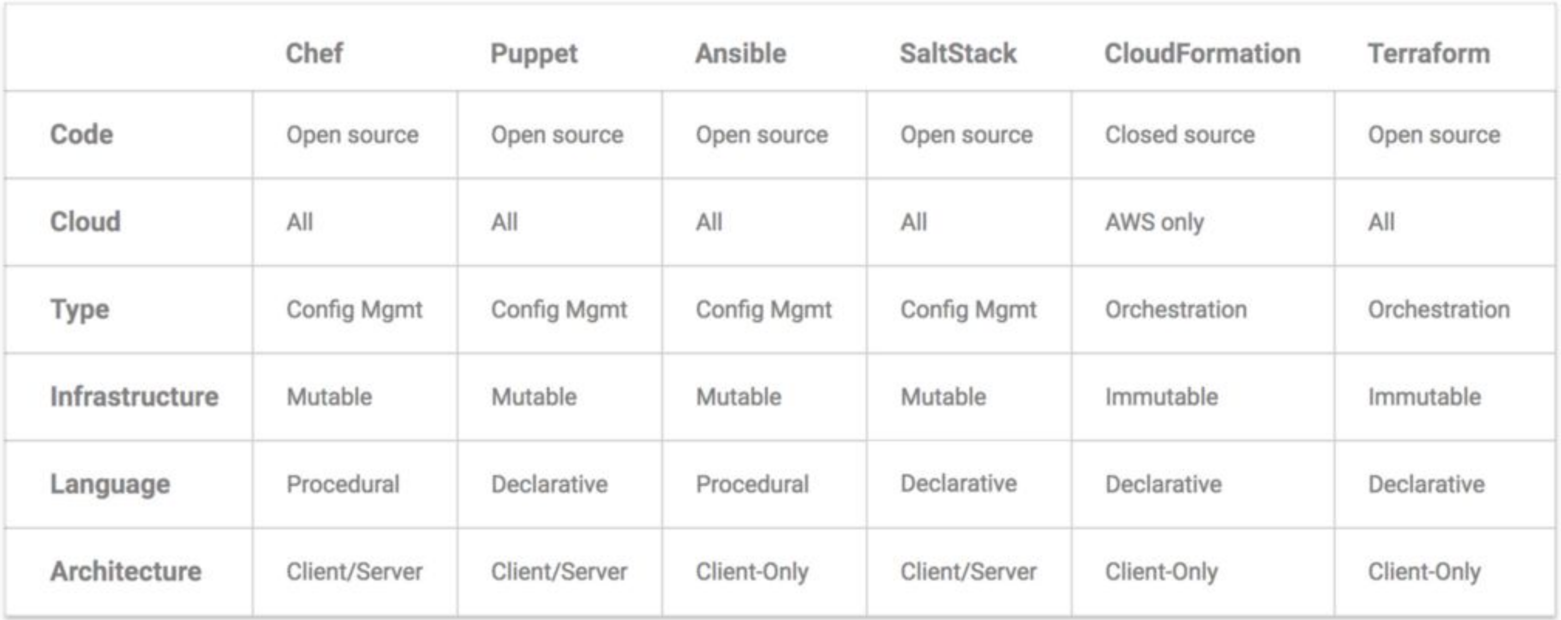
- Procedural vs Declarative Infrastructure
- Automation
- Agentcy
- Idempotency

Devops - Infrastructure as Code

- Non-functional requirements
  - Security
  - Backups
  - Availability
  - Upgradeability
  - Configuration mgmt
  - Monitoring
  - Logging
  - Metrics

SCALEOUT

Devops - Tools



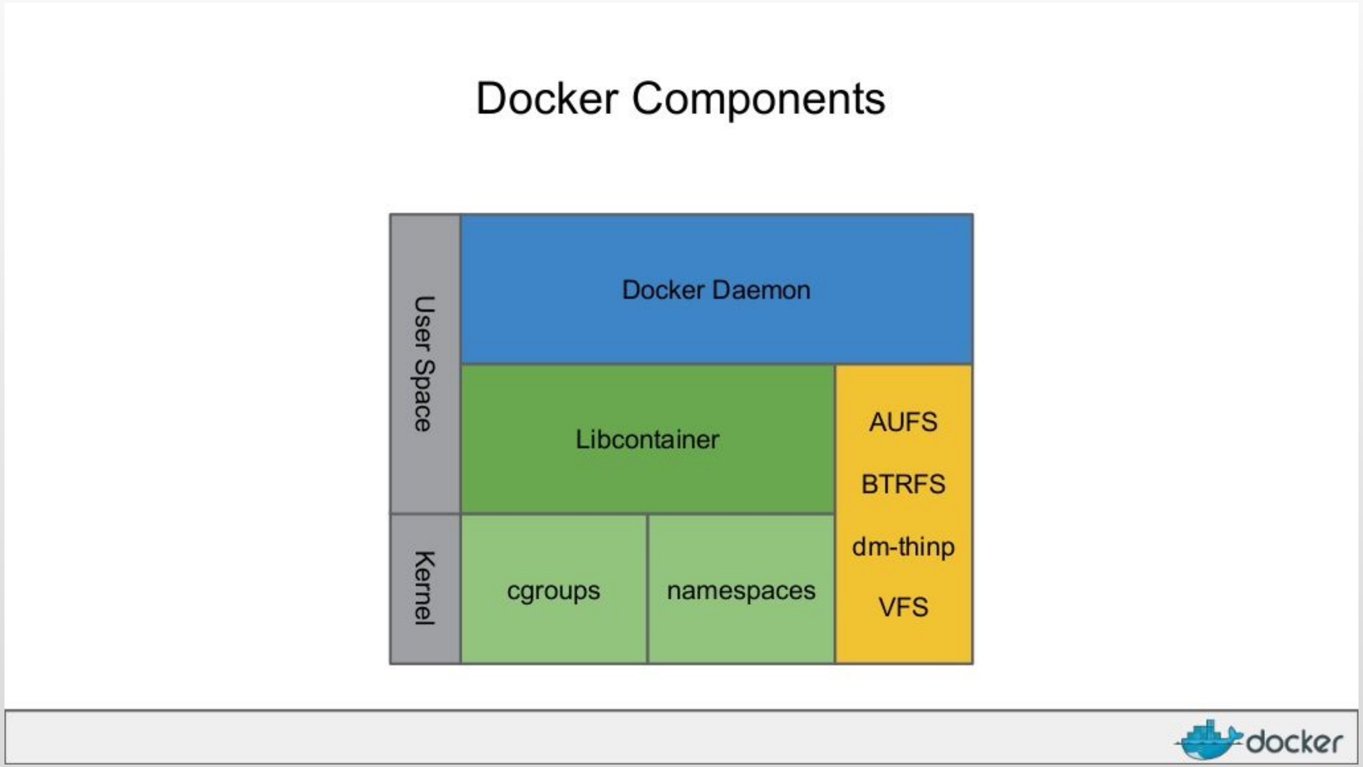
	Chef	Puppet	Ansible	SaltStack	CloudFormation	Terraform
Code	Open source	Open source	Open source	Open source	Closed source	Open source
Cloud	All	All	All	All	AWS only	All
Type	Config Mgmt	Config Mgmt	Config Mgmt	Config Mgmt	Orchestration	Orchestration
Infrastructure	Mutable	Mutable	Mutable	Mutable	Immutable	Immutable
Language	Procedural	Declarative	Procedural	Declarative	Declarative	Declarative
Architecture	Client/Server	Client/Server	Client-Only	Client/Server	Client-Only	Client-Only

SCALEOUT

Containers

- Containers share the host kernel
- Containers use the kernel ability to group processes for resource control
- Containers ensure isolation through namespaces
- Containers feel like lightweight VMs (lower footprint, faster), but are **not Virtual Machines!**

Docker



https://www.slideshare.net/RohitJnagal/docker-internals/4

Docker

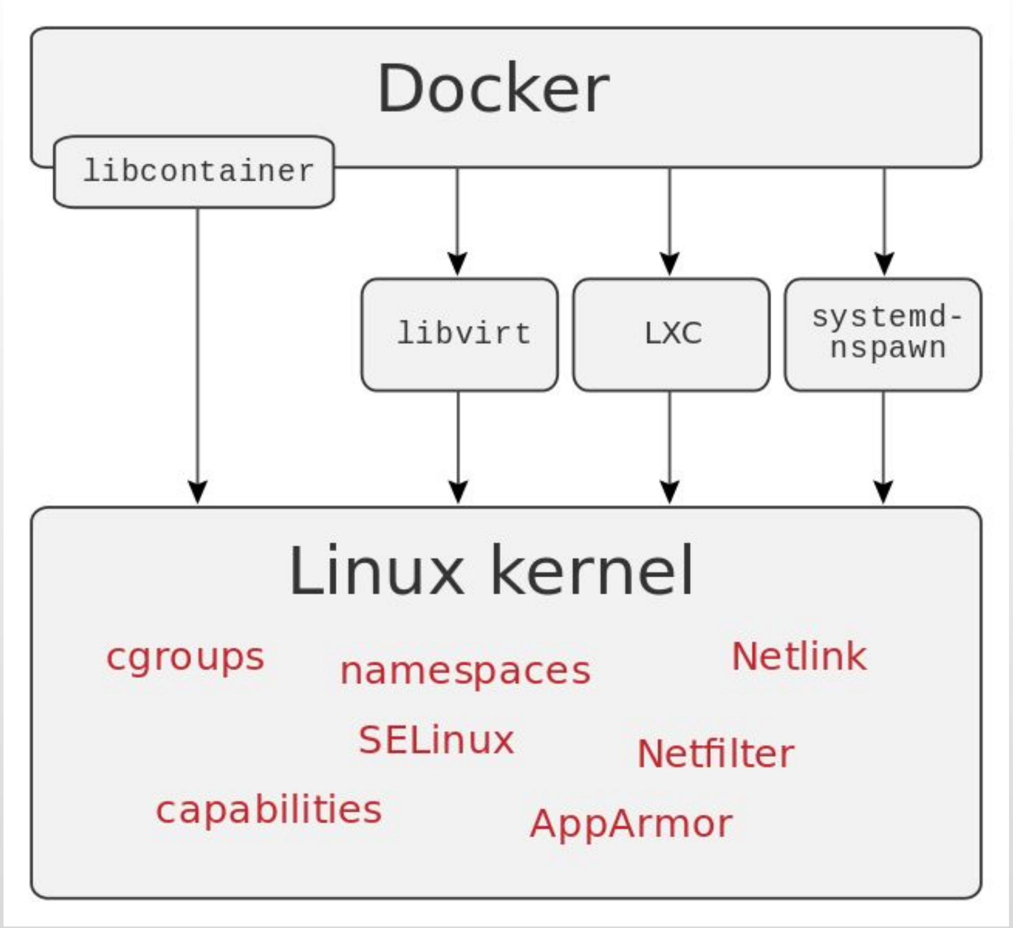


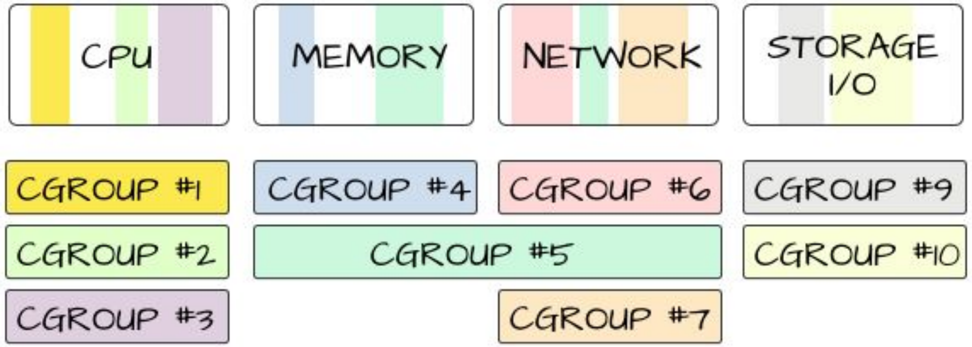
Image Source and Credits: https://delftswa.github.io/chapters/docker/

Docker

Docker Grounds up: Resource Isolation

Cgroups : Isolation and accounting

- cpu
- memory
- block i/o
- devices
- network
- numa
- freezer



http://www.slideshare.net/RohitJnagal/docker-internals and https://mairin.wordpress.com/2011/05/13/ideas-for-a-cgroups-ui/

SCALEOUT

Docker Basics commands

- Docker Run (--rm, -it, -d)
- Docker Start
- Docker image
- Docker Info
- Docker images / docker image ls
- Docker tag
- Docker rm
- Docker rmi
- Docker logs
- Docker build (-t [tag] .)
- Docker exec

Docker filtering information

- -- filter
- -- format
- Examples
  - docker ps --format "{{json .ID}}"
  - docker ps --format "{{ .ID }}"

Read More

- Cgroups https://mairin.wordpress.com/2011/05/13/ideas-for-a-cgroups-ui/
- Docker Internals: https://medium.com/@nagarwal/understanding-the-docker-internals-7ccb052ce9fe
- Api Gateway: https://microservices.io/patterns/apigateway.html

SCALEOUT