Day - 1

13 May 2025 18:27

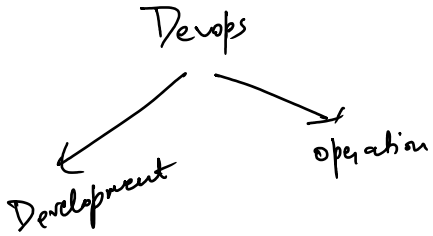
SRE - Site Reliability Engineer.

1. Devops

History!

What!

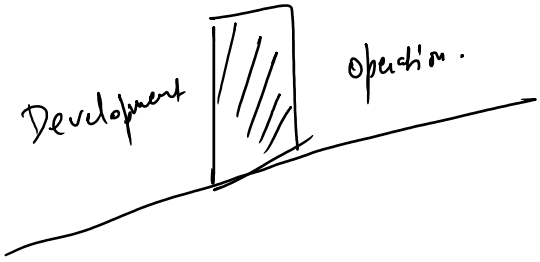
Use case.



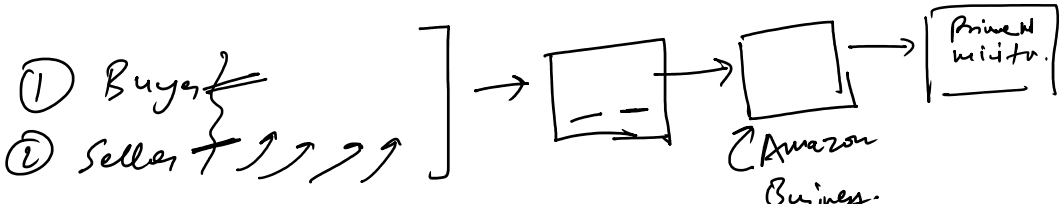
1. Pre Devops Era (Before 2007)
2. Birth of Devops (2007 - 2009)
3. Global Adoption & Tools Blooming (2010 - 2015)
4. Modern Devops Era (2016 - Present)

1. Pre Devops Era (Bfr 2007)

1. Development Team - Build feature, want feature to be live asap.
2. operation Team - uptime, reliable, accept user, stability



Traditional
1. Waterfall
2. Manual Testing & Deploy
3. (months/years) - long wait time.
4. Reactive incident handling.



② seller ↑↑↑↑ → Amazon Business.

② Birth of Devops (2007 - 2007) :-

① Developers & Sysadmin - voice dissatisfaction.

② Developers ↑ Operation → slow.

③ Global Adoption & Tools Boomy (2010 - 2015) :-

① Devops as philosphy.

② Focus - Automation, collaboration, monitoring, CI/CD and Infrastructure as code

① CI/CD Tools :- Jenkins, Travis CI

② Configuration Management :- Ansible, chef, Puppet.

③ Version Control :- Git, Gitlab, Github, Bitbucket, InnerSource.

④ Containerization :- Kubernetes (2014), Docker (2013), Openshift ↓ Redhat.

⑤ Infrastructure as Code :- Terraform (↑ HCL, ↑ json/yaml), Cloud formation (↑ json), Azure Resource Manager (ARM) (→ Python/yaml), Google cloud Deployment manager (CDM), Pulumi (→ Python/JS)

④ Modern Devops ERA (2016 - Present) :-

① Mainstream

② Rise of SRE (Site Reliability Engineers) from Google influenced by Devops.

Integrate :-

---

(1) AWS, Azure, GCP
(2.) Microservice & Kub. (K8)
(3.) Security (Dev Secops)

Finops, MLops, AIOps
Gitops

* Principle of Devops:-

(1) B/w Dev. & oper. team.
(2.) faster s/w Delivery.
(3.) Better Customer Satisfaction.

(1) Culture of Collaboration & Shared Responsibility:-

(1) Blameless post mortem.
(2.) Cross function team.
(3.) Shared goal & KPIs

(2.) Automation of ~~De~~ S/W Development Lifecycle

(1) Manual error reduces.
(2.) Consistency Dev, testing & Deployment.

CI/CD pipeline. - Automate Code Integration
Infrastructure As Code - Automate provisioning
Automated Testing - Unit, Integration & functional Tests

(3) Continuous Improvement & Feedback:-

→ Splunk, AppD, Dynatrace, ELK, Datadog.

(1) ... Tool (like machine APMs)

(3) [illegible] → Splunk, ELK, Datadog.

① Monitory Tool (logs, metrics, APMs)

② Users

③ Developer – Retrospective: feedback → Jira, Confluence, Customer Portal

① Monitory →

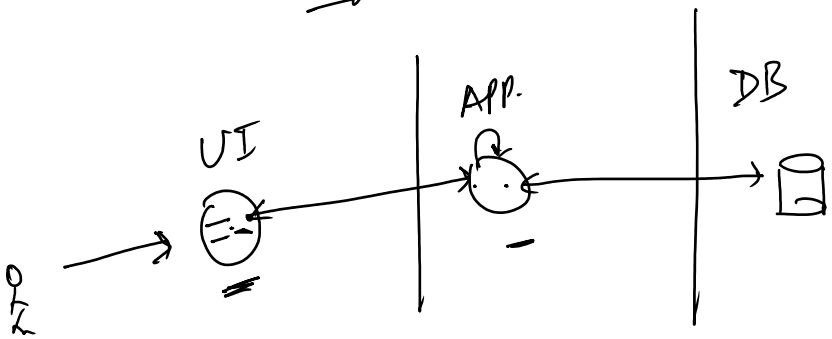
② observability →

Monitor → logs, Reactive

Observability → Proactive.

3 Pillar of observability:
① logs → Timestamp, ——
② Metrics → CPU = 30%, Memory = Critical
③ Traces

Login:



① Log → txt, json, XML

② Infra → CPU, Memory, Disk.

③ APM → Feature (Submit, Logout, Home) → Real → User Experience.

④ EUM (End user Monitor) → Synthetic → Bot (Hongkong)

(4) EUM (End user Monitoring) → Synthetic → [illegible]

(4) Continous Integration & Continous Delivery (CI/CD):-

(1) Frequently integrated into the main branch (CI)
(2) Automated Testing & deployed to production or staging (CD)

(1) Faster Release Cycle.
(2) Early bug Detection.
(3) Reduced Integration Risk.

(5) Monitoring & observability:-

Tools: Splunk, AppD, Dynatrace, New Relic, Datadog

(6) Security As a shared Responsibility (DevSecOps):-

(1) Static/Dynamic Code Analysis.
(2) Container Security.
(3) Secret Management (Vault, AWS Secret Managers)
(4) Compliance Check in CI/CD.

(7) Lean thinking & Elimination of Waste:-

(8) Infrastructure As Code (IaC):-

① Terraform.
② Cloudformation
③ ARM

⑨ Version Control:-

① Rollback
② Audit trails
③ Team Collaboration.

⑩ Resilience and Recovery Oriented Engineering:-

① chaos Engineering:- (Chaos Monkey)

* Devops Lifecycle:-

① Plan
② Develop
③ Build & Integrate
④ Test
⑤ Release
⑥ Deploy
⑦ Operate
⑧ Monitor & Observe
⑨ Feedback & Improvement

① Plan:- Business goal, Gather req. & plan the s/w Dev. lifecycle.

Jira, Azure Board, Trello, Confluence, Rally

② Develop:- Write, Build, & Compile source code.

Tools:- Git, Maven, Bitbucket, NPM, Ant,

2. Build & Integrate:-

① Bugs.

CI, Compile & Build, Code Coverage.

Tools:- ① Jenkins, GitHub, Circle CI, Travis CI

② SonarQube.

③ JUnit, NUnit, TestNG.

④ Test (Automated & Manual):-

Selenium, Junit, Postman, TestNG,
Jmeter, LoadRunner, Burpsites.

⑤ Release (CD):-

Production or staging environment

① DockerHUB, JFrog Artifactory, Nexus, Spinnaker, Argo CD

⑥ Deploy:-

Tools:-

① Kubernetes, Docker, Terraform, Ansible

② AWS CodeDeploy, OctopusDeploy, AzureDevops.

⑦ Operate:- Maintain Infrastructure (s/w availability, reliability & Performance.

Tools:- ① Nagios, Zabbix, Graphana.

② NewRelic, Datadog, Dynatrace.

ServiceNow.

(1) ...

(2) New Relic, ...

(3) PagerDuty, ServiceNow.

(8) Monitor & observe

Splunk ELK, opentelemetry.

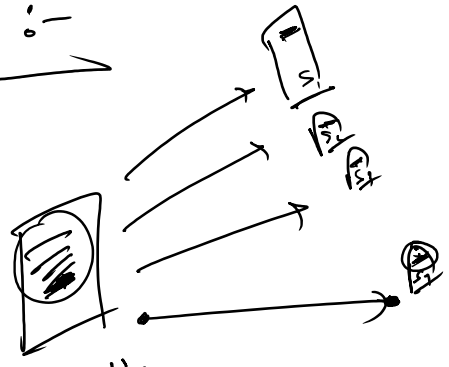
(9) Feedback & Improve

Metrics, Incidant & User feedback.

Tools:- Confluence, slack, jira

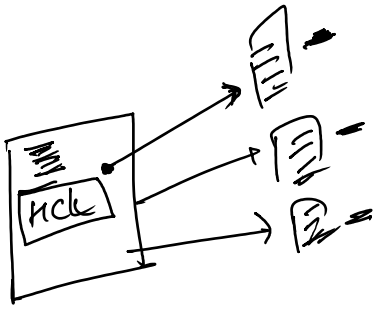
Gitops & Infrastructure as Code (Iac) :-

Iac (Infrastructure as Code) :-



Controller node

Ansible/puppet



Server.

Terraform

azure | gcp

AWS | Account.

```
provider "aws" {
  region  = "us-east-1"
  profile = "default" # Optional: depends on your AWS CLI config
}

resource "aws_instance" "example" {
```

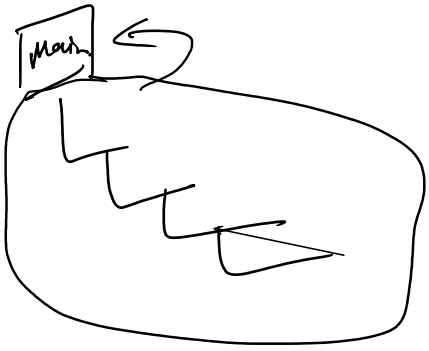
```
  ami           = "ami-0c55b159cbfafe1f0"  # Example Amazon Linux 2 AMI
  instance_type = "t2.micro"

  tags = {
    Name = "Terraform-EC2"
  }
}
```

Iac lifecycle

Gitops:- App. & Infra Deployment.

Automate provisioning, rollback, reconciliation.



1. Pull Request (PR)
2. CI pipeline runtest & validate.
3. Merge Main Branch.
4. Gitops (Argo CD, Flux)
5. cluster / environment.
6.