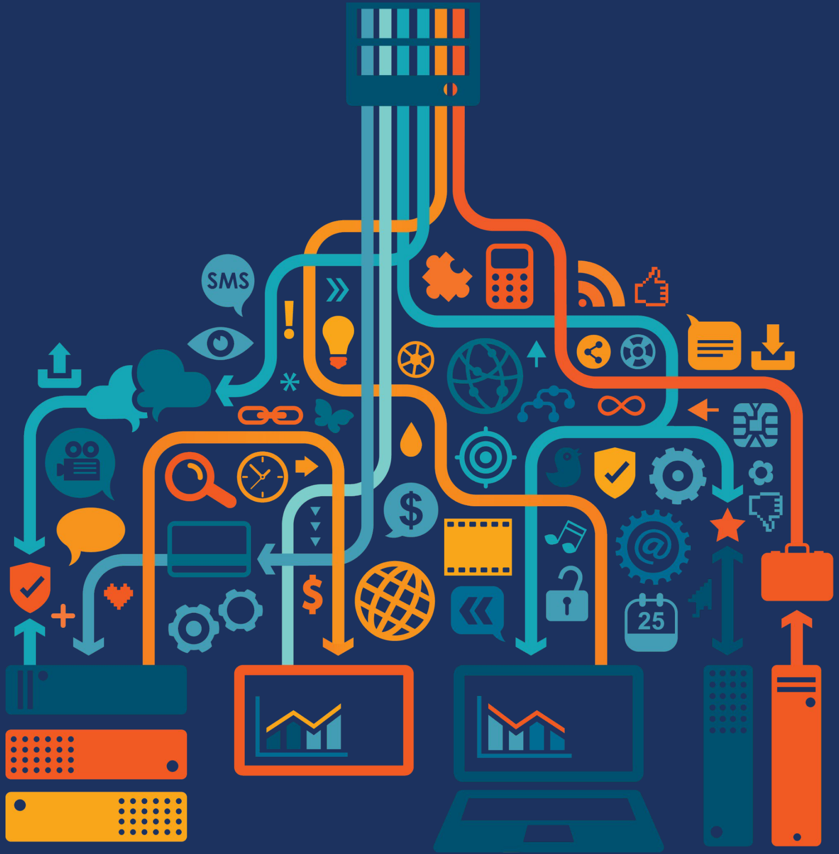
DevOps Bootcamp

ITJ BOOTCAMP



HELLO!

I am Erik Flores

I am here because I am passionate about how DevOps can enable Dev teams deliver faster and more reliable software.

DevOps Introduction

Agenda

- Origins
- Approach to DevOps
- Benefits
- Roles
- Skills
- Reads
- Tech Debt
- DevOps Health Radar

1. Origins: Teams involved in SDLC

DEVS

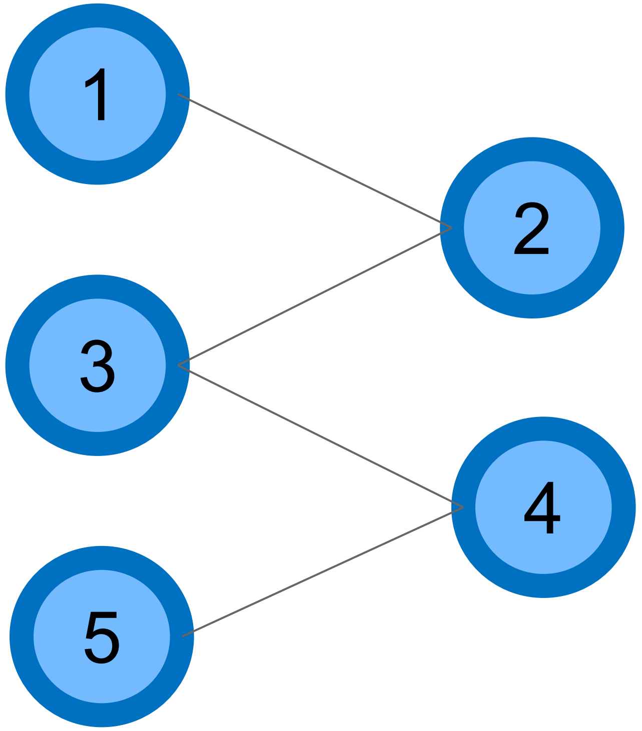
Focus on coding and creating new functionality.
Goal: Meet Biz/mkt demands.
Faster/Fiercer competition.

INFRA

Purchase, configure and maintain servers.
Old-school budget and process constraints.

SEC

Protect digital security and/or regulation.
Old-school left until the end of the SDLC.



QA/TEST

Will test Dev integrations in a separate environment and report any bugs back to dev.

OPS

Ensure production is running smoothly, report on any failures. Old-school own logs.
Goal: Stability.

1. Origins: Conflicting Interests

DEVS VS OPS:
CREATES SILOS

- Devs pushing for changes asap
- Ops looking for stability
- Teams are not aligned and don’t work well together.

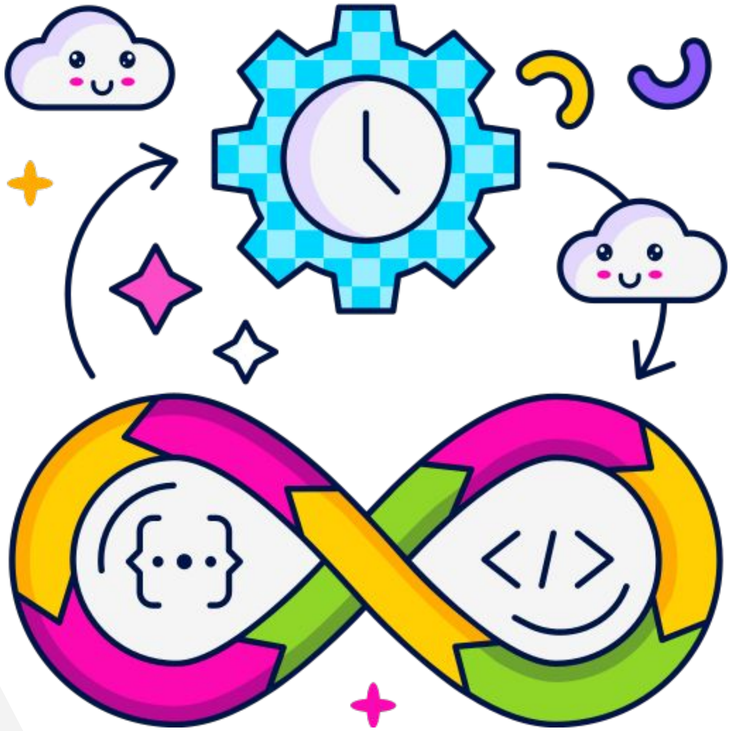
1. Origins: DevOps is Born

PLAN

CODE

BUILD

TEST



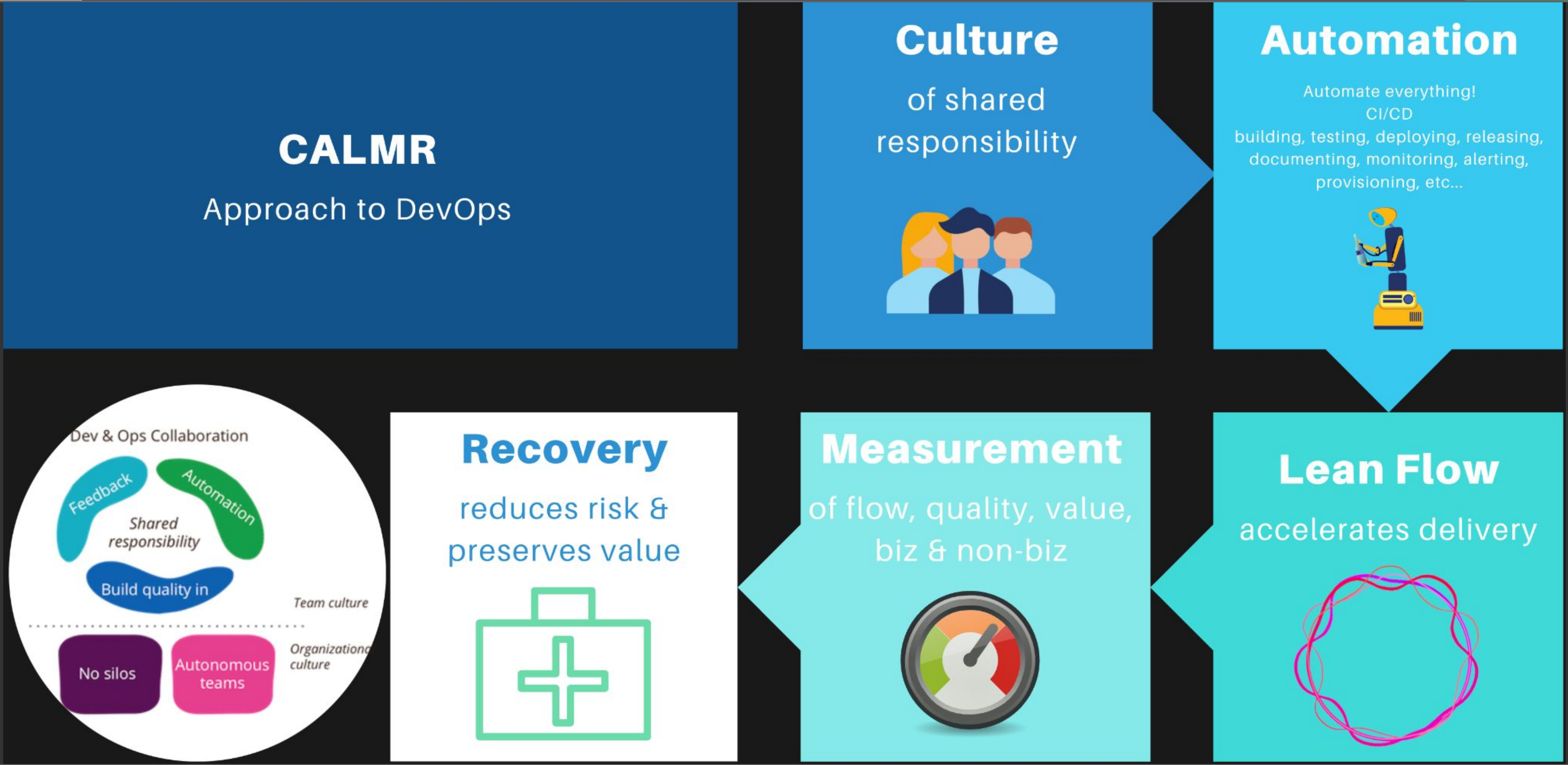
RELEASE

DEPLOY

OPERATE

MONITOR

2: Approach to DevOps



CALMR

Approach to DevOps

Culture

of shared responsibility

Automation

Automate everything!
CI/CD
building, testing, deploying, releasing, documenting, monitoring, alerting, provisioning, etc...

Lean Flow

accelerates delivery

Measurement

of flow, quality, value, biz & non-biz

Recovery

reduces risk & preserves value

https://martinfowler.com/bliki/DevOpsCulture.html https://www.scaledagileframework.com/calmr/

3. Benefits

440 X
Faster Lead time (start to finish)

96 X
Fewer Defects

24 X
Faster Recovery

46 X
More frequent deployments

29 X
More time spent on NEW work

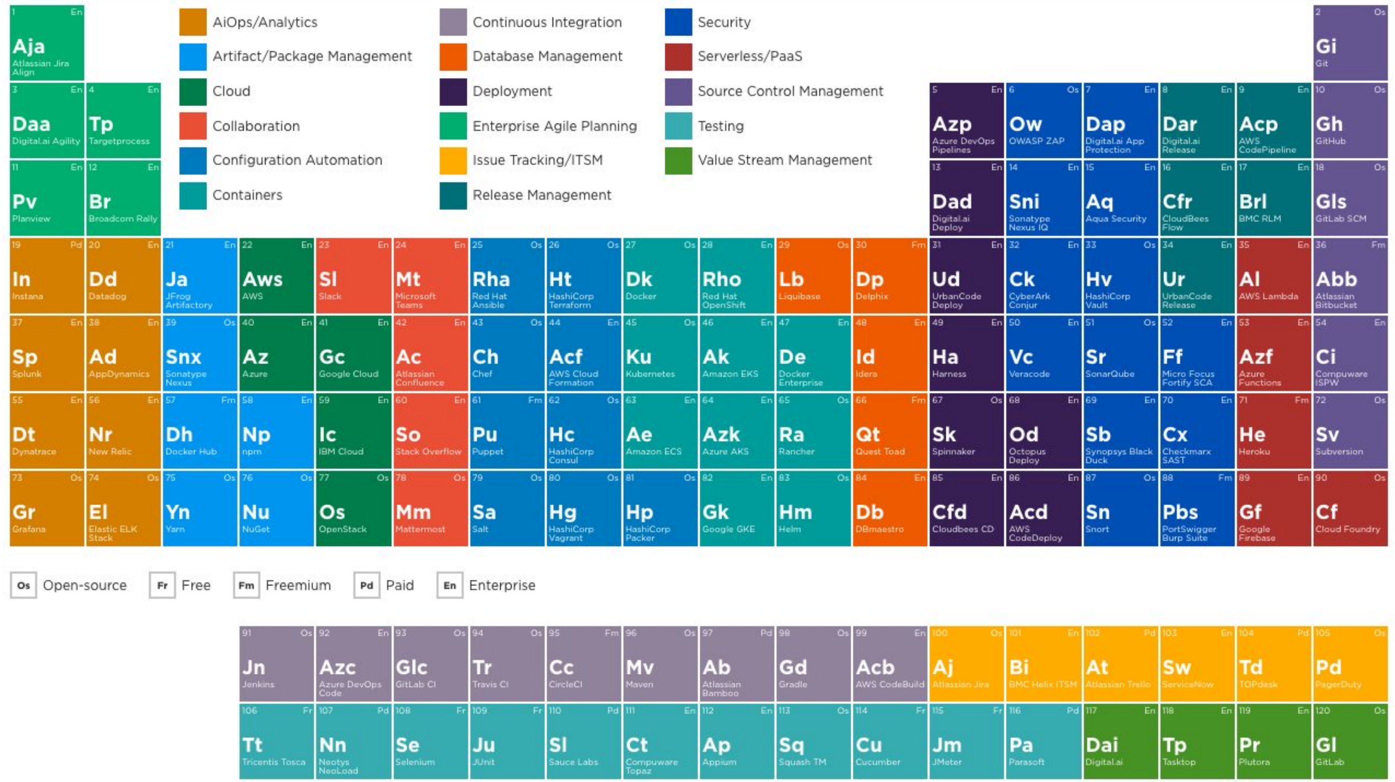
2 X
Less time spent fixing security issues

2.2 X
improved NPS due to less burnout

2 X
More likely to exceed profitability goals

Source: https://puppet.com/resources/whitepaper/state-of-devops-report

4. Roles



- AiOps/Analytics
- Artifact/Package Management
- Cloud
- Collaboration
- Configuration Automation
- Containers
- Continuous Integration
- Database Management
- Deployment
- Enterprise Agile Planning
- Issue Tracking/ITSM
- Release Management
- Security
- Serverless/PaaS
- Source Control Management
- Testing
- Value Stream Management

1 En **Aja** Atlassian Jira Align																	2 Os **Gi** Git
3 En **Daa** Digital.ai Agility	4 En **Tp** Targetprocess											5 En **Azp** Azure DevOps Pipelines	6 Os **Ow** OWASP ZAP	7 En **Dap** Digital.ai App Protection	8 En **Dar** Digital.ai Release	9 En **Acp** AWS CodePipeline	10 Os **Gh** GitHub
11 En **Pv** Planview	12 En **Br** Broadcom Rally											13 En **Dad** Digital.ai Deploy	14 En **Sni** Sonatype Nexus IQ	15 En **Aq** Aqua Security	16 En **Cfr** CloudBees Flow	17 En **Brl** BMC RLM	18 Os **Gls** GitLab SCM
19 Pd **In** Instana	20 En **Dd** Datadog	21 En **Ja** JFrog Artifactory	22 En **Aws** AWS	23 En **Sl** Slack	24 En **Mt** Microsoft Teams	25 Os **Rha** Red Hat Ansible	26 Os **Ht** HashiCorp Terraform	27 Os **Dk** Docker	28 En **Rho** Red Hat OpenShift	29 Os **Lb** Liquibase	30 Fm **Dp** Delphix	31 En **Ud** UrbanCode Deploy	32 En **Ck** CyberArk Conjur	33 Os **Hv** HashiCorp Vault	34 En **Ur** UrbanCode Release	35 En **Al** AWS Lambda	36 Fm **Abb** Atlassian Bitbucket
37 En **Sp** Splunk	38 En **Ad** AppDynamics	39 Os **Snx** Sonatype Nexus	40 En **Az** Azure	41 En **Gc** Google Cloud	42 En **Ac** Atlassian Confluence	43 Os **Ch** Chef	44 En **Acf** AWS Cloud Formation	45 Os **Ku** Kubernetes	46 En **Ak** Amazon EKS	47 En **De** Docker Enterprise	48 En **Id** Idera	49 En **Ha** Harness	50 En **Vc** Veracode	51 Os **Sr** SonarQube	52 En **Ff** Micro Focus Fortify SCA	53 En **Azf** Azure Functions	54 En **Ci** Compuware ISPW
55 En **Dt** Dynatrace	56 En **Nr** New Relic	57 Fm **Dh** Docker Hub	58 En **Np** npm	59 En **Ic** IBM Cloud	60 En **So** Stack Overflow	61 Fm **Pu** Puppet	62 Os **Hc** HashiCorp Consul	63 En **Ae** Amazon ECS	64 En **Azk** Azure AKS	65 Os **Ra** Rancher	66 Fm **Qt** Quest Toad	67 Os **Sk** Spinnaker	68 En **Od** Octopus Deploy	69 En **Sb** Synopsys Black Duck	70 En **Cx** Checkmarx SAST	71 Fm **He** Heroku	72 Os **Sv** Subversion
73 Os **Gr** Grafana	74 Os **El** Elastic ELK Stack	75 Os **Yn** Yarn	76 Os **Nu** NuGet	77 Os **Os** OpenStack	78 Os **Mm** Mattermost	79 Os **Sa** Salt	80 Os **Hg** HashiCorp Vagrant	81 Os **Hp** HashiCorp Packer	82 En **Gk** Google GKE	83 Os **Hm** Helm	84 En **Db** DBmaestro	85 En **Cfd** Cloudbees CD	86 En **Acd** AWS CodeDeploy	87 Os **Sn** Snort	88 Fm **Pbs** PortSwigger Burp Suite	89 En **Gf** Google Firebase	90 Os **Cf** Cloud Foundry

Os Open-source Fr Free Fm Freemium Pd Paid En Enterprise

91 Os **Jn** Jenkins	92 En **Azc** Azure DevOps Code	93 Os **Glc** GitLab CI	94 Os **Tr** Travis CI	95 Fm **Cc** CircleCI	96 Os **Mv** Maven	97 Pd **Ab** Atlassian Bamboo	98 Os **Gd** Gradle	99 En **Acb** AWS CodeBuild	100 Os **Aj** Atlassian Jira	101 En **Bi** BMC Helix ITSM	102 Pd **At** Atlassian Trello	103 En **Sw** ServiceNow	104 Pd **Td** TOPdesk	105 Os **Pd** PagerDuty
106 Fr **Tt** Tricentis Tosca	107 Pd **Nn** Neotys NeoLoad	108 Fr **Se** Selenium	109 Fr **Ju** JUnit	110 Pd **Sl** Sauce Labs	111 En **Ct** Compuware Topaz	112 En **Ap** Appium	113 Os **Sq** Squash TM	114 Fr **Cu** Cucumber	115 Fr **Jm** JMeter	116 Pd **Pa** Parasoft	117 En **Dai** Digital.ai	118 En **Tp** Tasktop	119 En **Pr** Plutora	120 Os **Gl** GitLab

https://digital.ai/periodic-table-of-devops-tools

4. Roles

SCM

Software Configuration Management:
Helps configure and integrate tools into the CI/CD pipeline

+

EM

Environment Management:
Provide IaC for timely and precise infrastructure delivery

+

SRE

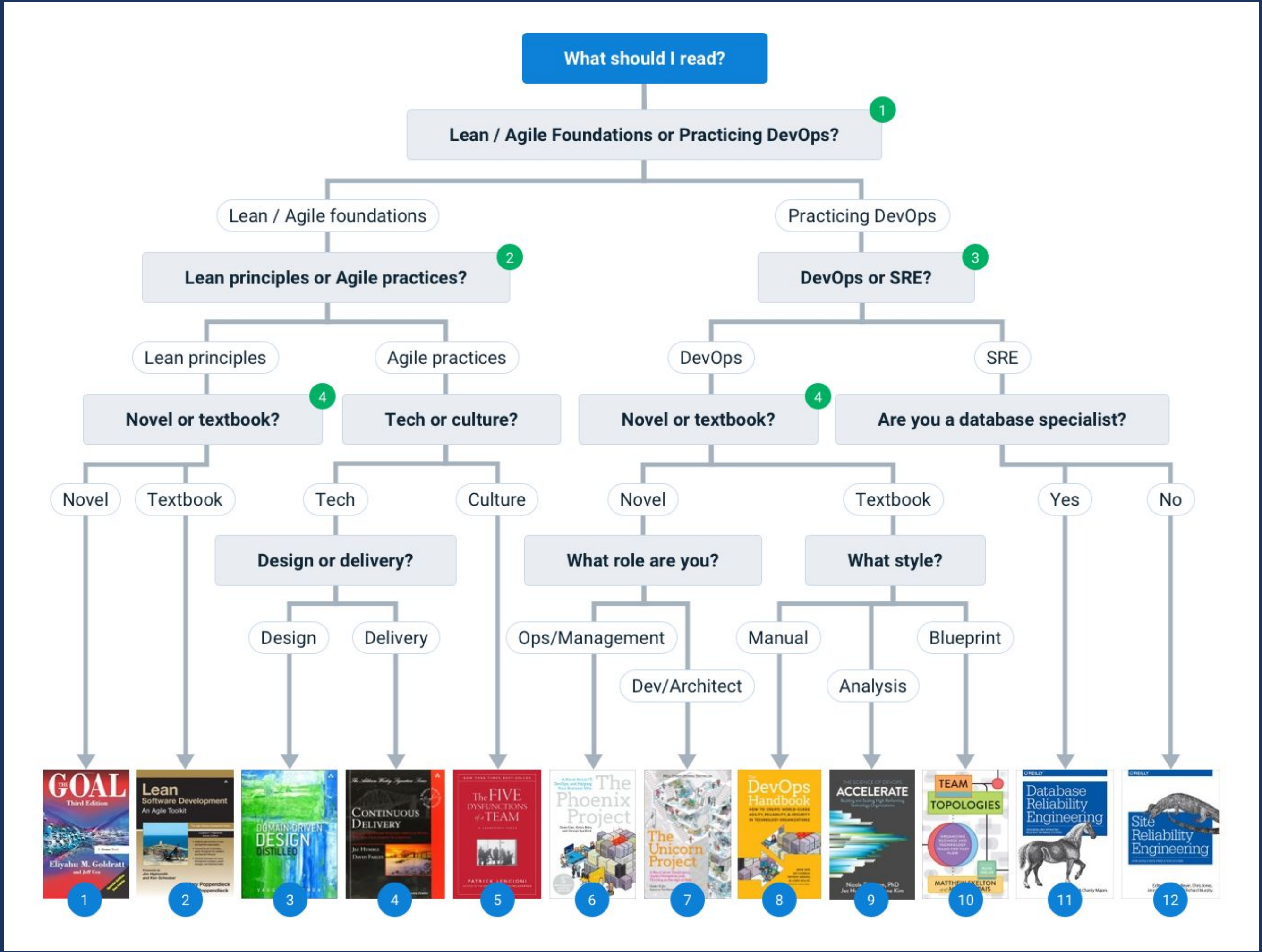
Site Reliability Engineering
Designs and implements observability solutions and self-healing capabilities

DevOps Areas

You can specialize on any of the following areas, or become a “full-stack” DevOps Engineer.

5. Skills

- Separate deploy and release
- Architect for operations (telemetry for biz/tech, logging, scaling, microservices, decoupling)
- Thread Modeling (shift left on security)
- Build and test automation
- Gated Commits
- Visualize and monitor the build and test process
- Avoid long lived branches
- Environment congruity
- Test automation: functional, integration, regression, performance, security, penetration,
- Test data mgt
- Environment virtualization, Infrastructure as Code
- Blue/Green deployments
- Dark launches
- Feature Toggles
- Deployment Automation
- Selective deployments (to target customers by geography for ex)
- Self-service deployments
- Production testing via automation (smoke tests)
- Full stack telemetry, visual displays
- AIOps
- Self-healing
- Chaos engineering
- Session replay
- Immutable infrastructure (no changes directly in prod)

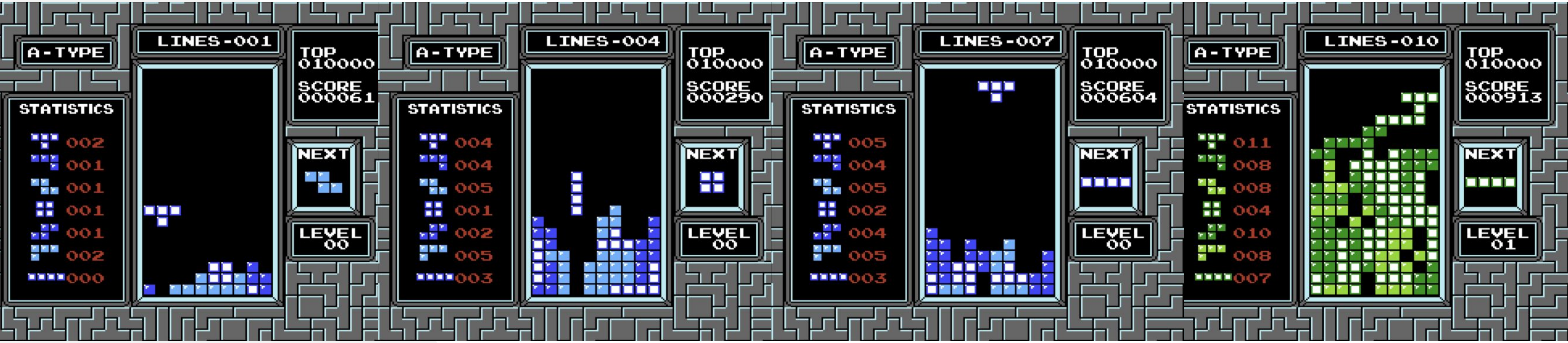

What should I read?
1
Lean / Agile Foundations or Practicing DevOps?
Lean / Agile foundations
Practicing DevOps
2
Lean principles or Agile practices?
3
DevOps or SRE?
Lean principles
Agile practices
DevOps
SRE
4
Novel or textbook?
Tech or culture?
4
Novel or textbook?
Are you a database specialist?
Novel
Textbook
Tech
Culture
Novel
Textbook
Yes
No
Design or delivery?
What role are you?
What style?
Design
Delivery
Ops/Management
Dev/Architect
Manual
Analysis
Blueprint
THE GOAL
Third Edition
Eliyahu M. Goldratt and Jeff Cox
1
Lean Software Development
An Agile Toolkit
Poppendieck
2
DOMAIN-DRIVEN DESIGN DISTILLED
3
Continuous Delivery
Jez Humble
David Farley
4
The FIVE DYSFUNCTIONS of a TEAM
PATRICK LENCIONI
5
The Phoenix Project
6
The Unicorn Project
7
The DevOps Handbook
8
ACCELERATE
9
TEAM TOPOLOGIES
10
Database Reliability Engineering
11
Site Reliability Engineering
12

Interestingly, Satya Nadella, CEO of Microsoft, still has a culture that if a developer has a choice between working on a feature or developer productivity, they should always choose developer productivity.

-Gene Kim, The Phoenix Project

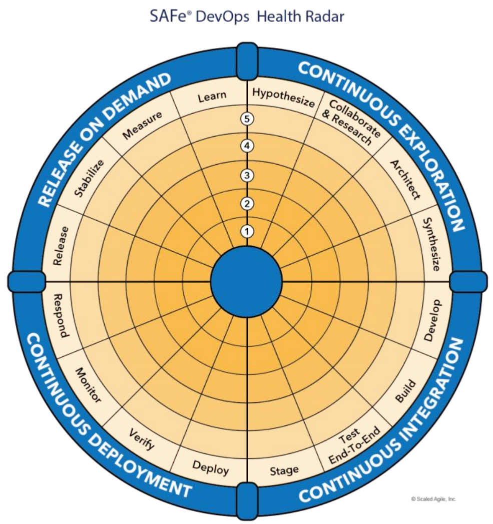
7. Tech Debt

https://medium.com/s/story/technical-debt-is-like-tetris-168f64d8b700



8. DevOps Health Radar

Sit	Fly
The team backlog does not exist or is not used to manage daily work.	**Code is checked in multiple times per day;** tests are written before code (TDD); pair work and other Built-in quality practices are the norm.
Builds are run fewer than once per iteration and/or are completely manual.	**Builds run on every commit;** builds include static code analysis and security testing; gated commits prevent defects from entering the version control; dev branches are merged to trunk on every commit."
Testing is performed manually in environments that do not mimic production; testing occurs in large batches during a scheduled "testing" phase.	**Successful builds trigger automatic deployment to production-like test environments**; all tests are automated; tests run in parallel and changes are fully validated after every commit.
No staging environment exists or we use a test environment for staging.	Stories, changes and infrastructure are auto-deployed to a staging environment, validated, and immediately proceed to deployment.
Features are deployed to production every 3+ months; deployments are manual and painful; "deployed" implies "released".	**Features are deployed continuously throughout each iteration;** Dev teams initiate deployments directly via pipeline tools; release is completely decoupled from deployment; dark releases are the norm.
Deployments are not verified in production before being released to end users.	Automated production tests run on an ongoing basis and feed monitoring systems; f**ailed deployments can be rolled back instantly or fixed forward through the entire pipeline.**
No feature level production monitoring exists; only infrastructure monitoring is in place.	Federated monitoring platform provides **one-stop access to full-stack insights**; data is used to gauge system performance and business value.
Customers find issues before we do; resolving high priority issues is time consuming and reactive; customers have low confidence in our ability to recover from production issues.	Our monitoring systems alert us to dangerous conditions based on carefully-designed **tolerance thresholds**; **Developers are responsible for supporting their own code** and proactively issue fixes through the pipeline before users are affected.



BOOTCAMP

THANKS!

Any questions?

You can find me at

/in/erikfloresv