ATLASSIAN

The content described herein is intended to outline our general product direction for informational purposes only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described herein remain at the sole discretion of Atlassian and is subject to change.

Breaking all the DevOps rules

Mirco Hering |@mircohering

accenture

This is the hard reality in many large enterprises

A few years ago, I would have been disappointed with the following statements:

- Our team does not fail pipelines with failed tests.
- Our team does deploy code that has failed security scans.
- Our team chooses manual cloud deployments over automation.
- Our team moves from full stack to a federated model.

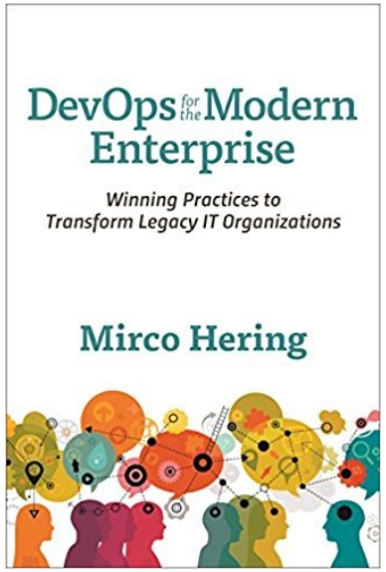
Hello, my name is Mirco Hering

Mirco Hering
Global DevOps Lead

And I sometimes break rules…

Blog at http://notafactoryanymore.com

…consciously.



The environment we're talking about is that of larger complex enterprises

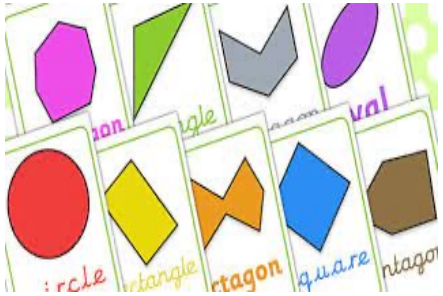
The transformation starting point is along the following lines:

Portfolio

Complex contract

Lack of documentation

Infrastructure

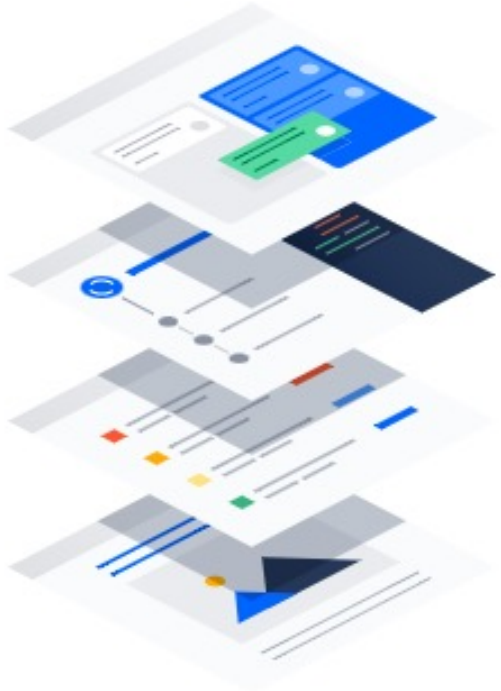


Custom, COTS, SAAS

BAs, testers, engineers, etc.

The Atlassian landscape we work with

- Jira Software
- Bitbucket
- Opsgenie
- Confluence



Our ground to cover

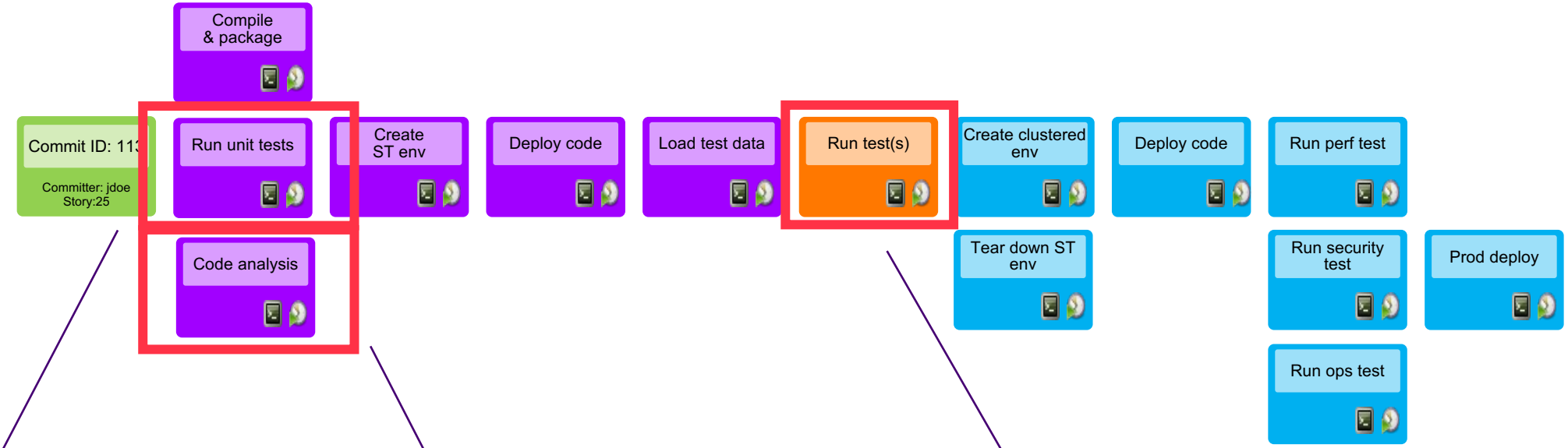
1. CD pipelines
2. Full stack teams
3. The cloud

→ And of course, some take-home tips!

Pipelines — good intentions

"The goal of CD pipelines is to build/deploy/assess each new version of an application and provide fast, comprehensive feedback."

Pipelines — from pipelines to networks

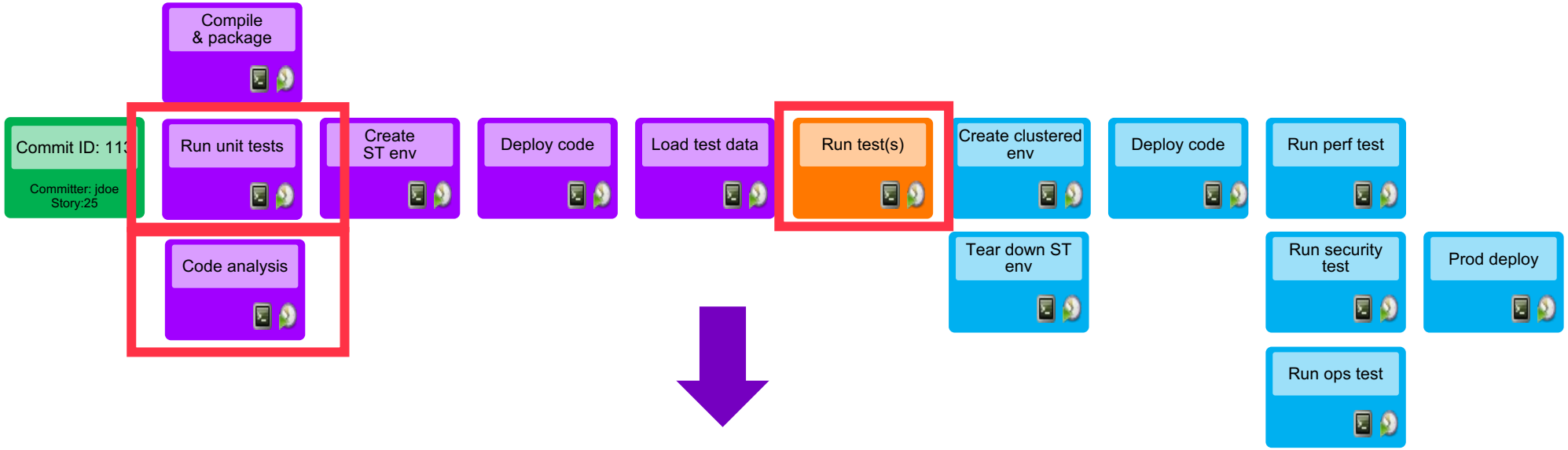


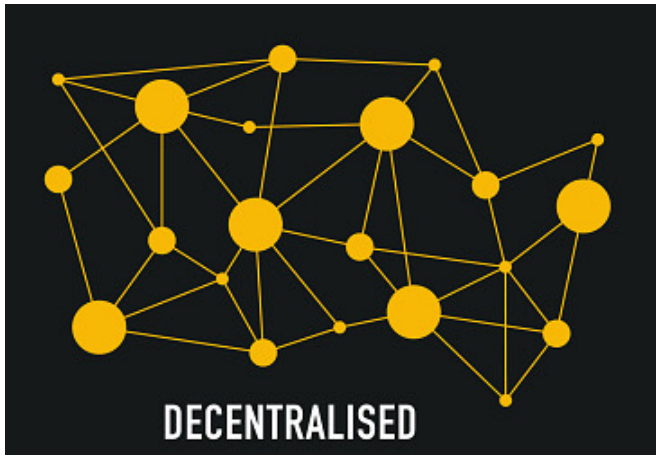
- Frequency of usage
- Maintenance

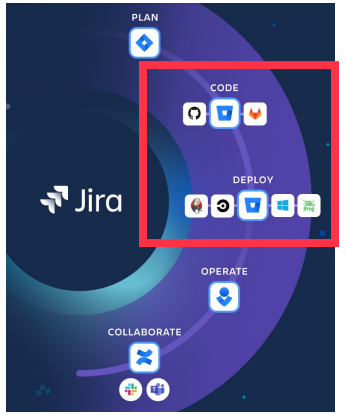
- Impact assessment
- Duration
- Risk velocity

- Frequency
- Cost
- Maintenance
- License

Pipelines — from pipelines to networks



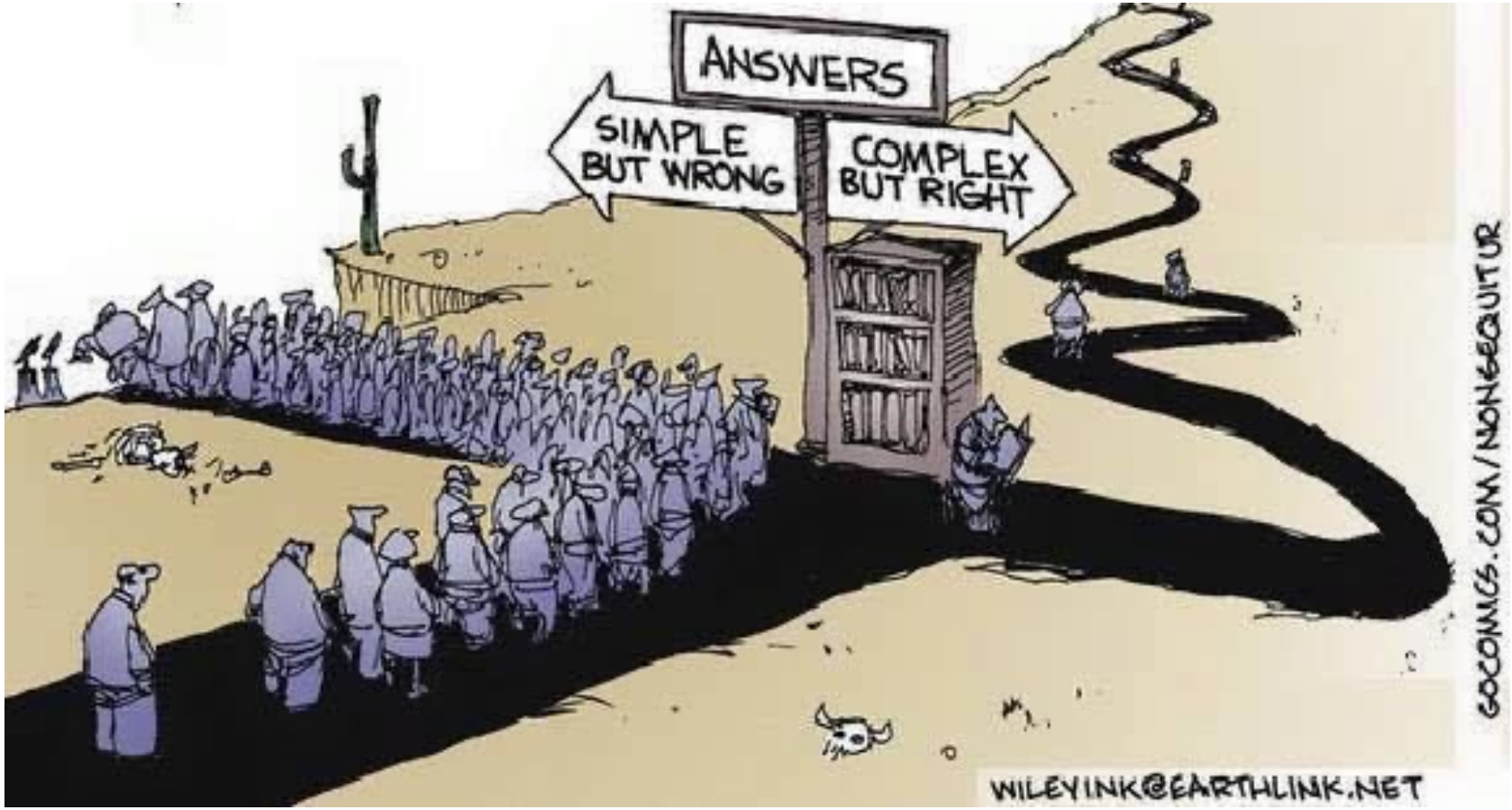




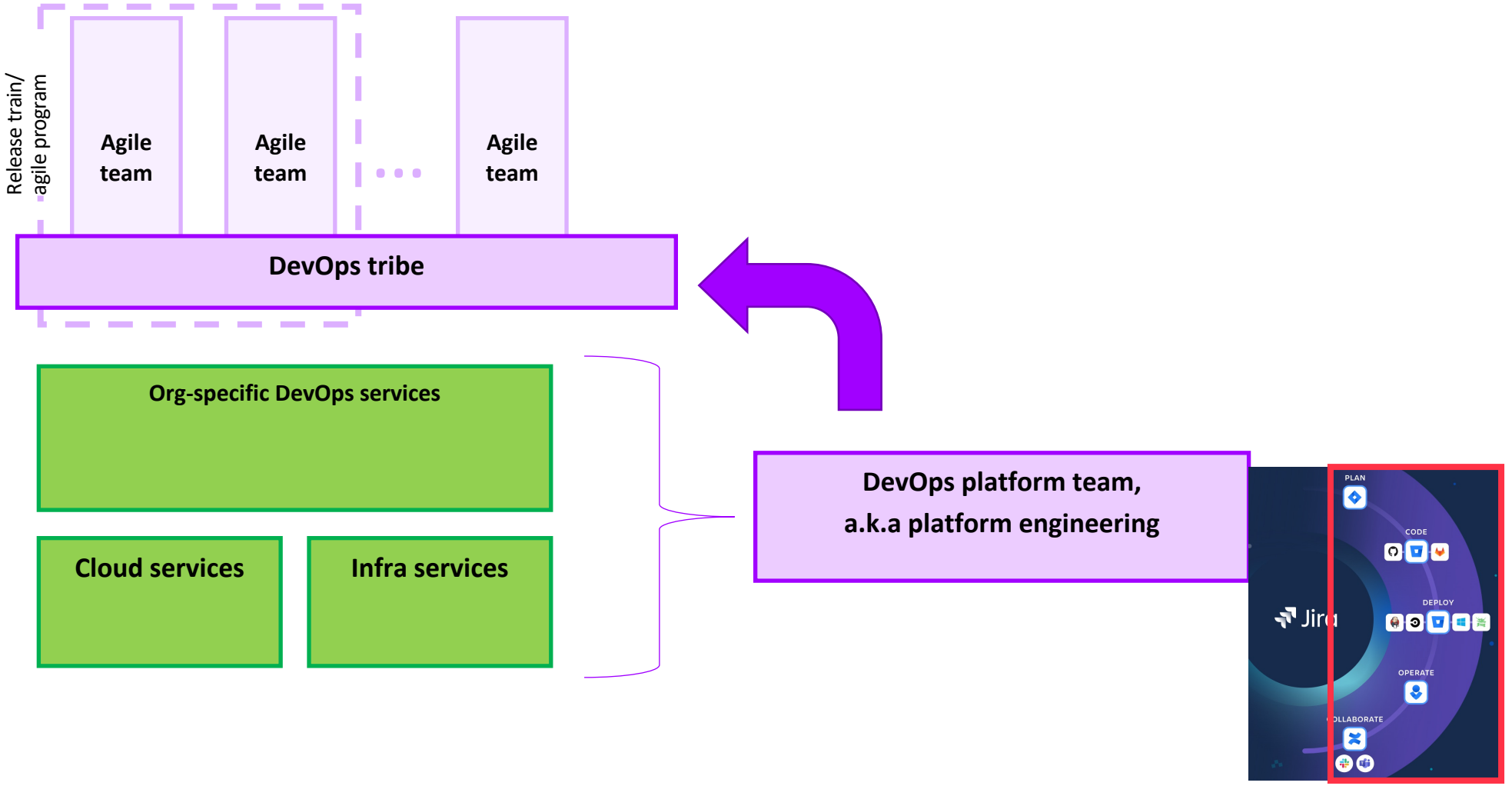
Full stack teams — good intentions

"The most efficient way for DevOps to organize is in a full stack team that includes people from the infrastructure team."

Hmm…

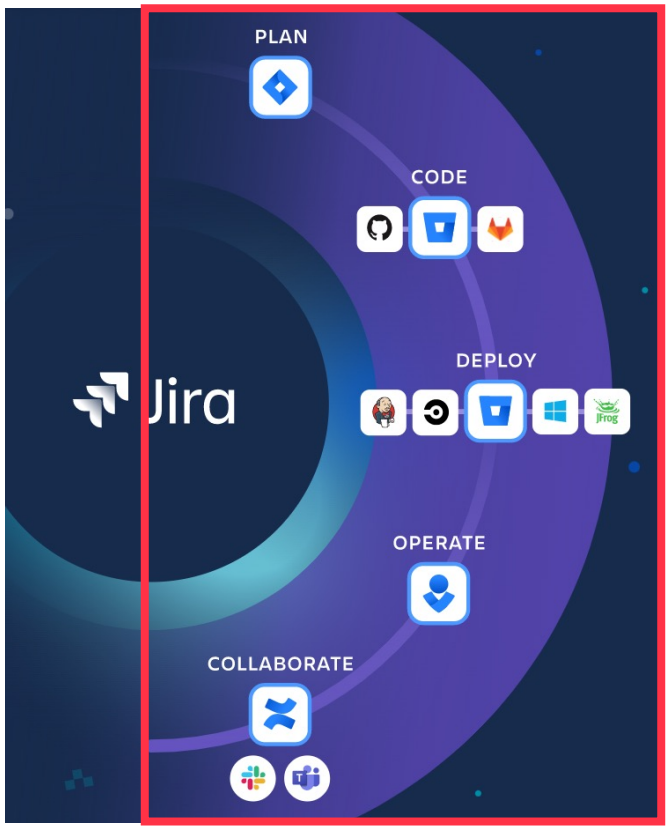


To DevOps team or not to DevOps team?



Creating autonomous teams instead of full stack teams

1. Make information available (e.g., log aggregation)
2. Provide self-service for the obvious (passwords, servers, etc.)
3. Measure dependencies and eradicate them



Question: Is the team ready for full autonomy?

Hmm…

Cloud — good intentions

"Environments in the cloud are full automated, and nothing should be manual."

and…

"In the cloud, servers are immutable."

Cloud — the better way

1

Experiment

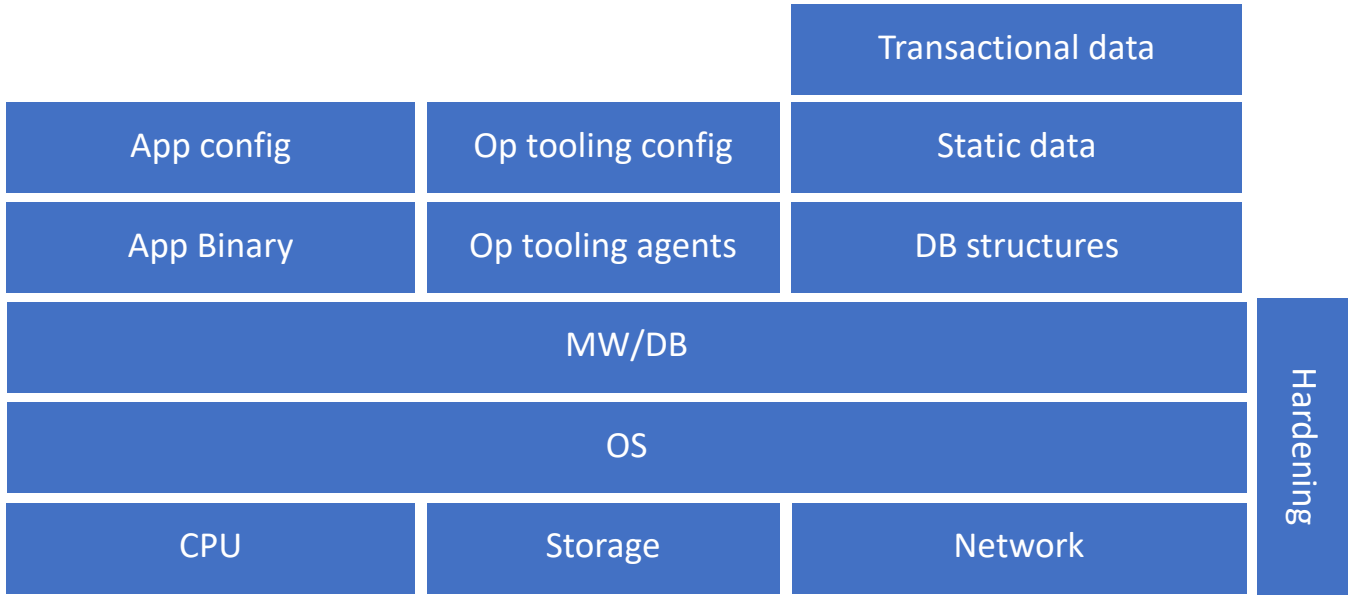
2

Describe

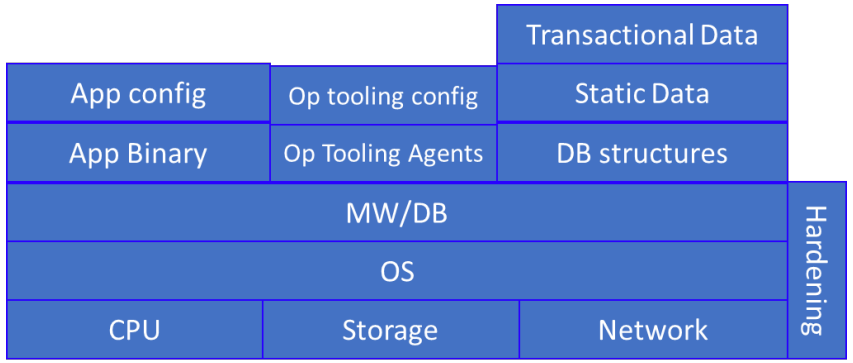
3

Automate

Cloud golden image vs. automation in layers



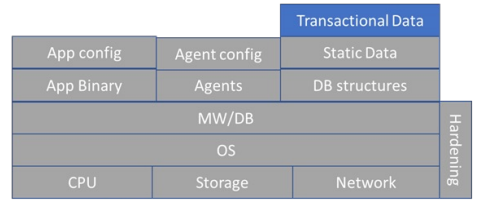
Cloud golden image vs. automation in layers



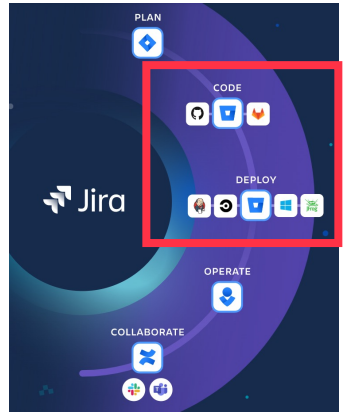
1 Provision Image

2 Apply IA pattern

3 Deploy Application



4 Load Data



The boardroom question

How do I know we are on the right track if each answer is contextual?

It depends.

It depends.

It depends.

It depends.

It depends.

It depends.

It depends.

It depends.

It depends.

It depends.

It depends.

It depends.

Rule 1: Make economic decisions

Rule 2: Measure things

Business effectiveness
Experiment & expand

Delivery efficiency
Speed up & automate

Operations reliability
Automate & react

Architectural flexibility
Simplify & decouple

Rule 2: Measure things

Business effectiveness | Experiment & expand

- Number of experiments run
- Average time to decision
- Funnel data
- Number of systems interactions
- Feature usage

Delivery efficiency | Speed up & automate

- Cycle time
- Number of manual steps
- Number of deployment failures
- Time to feedback
- Transaction cost

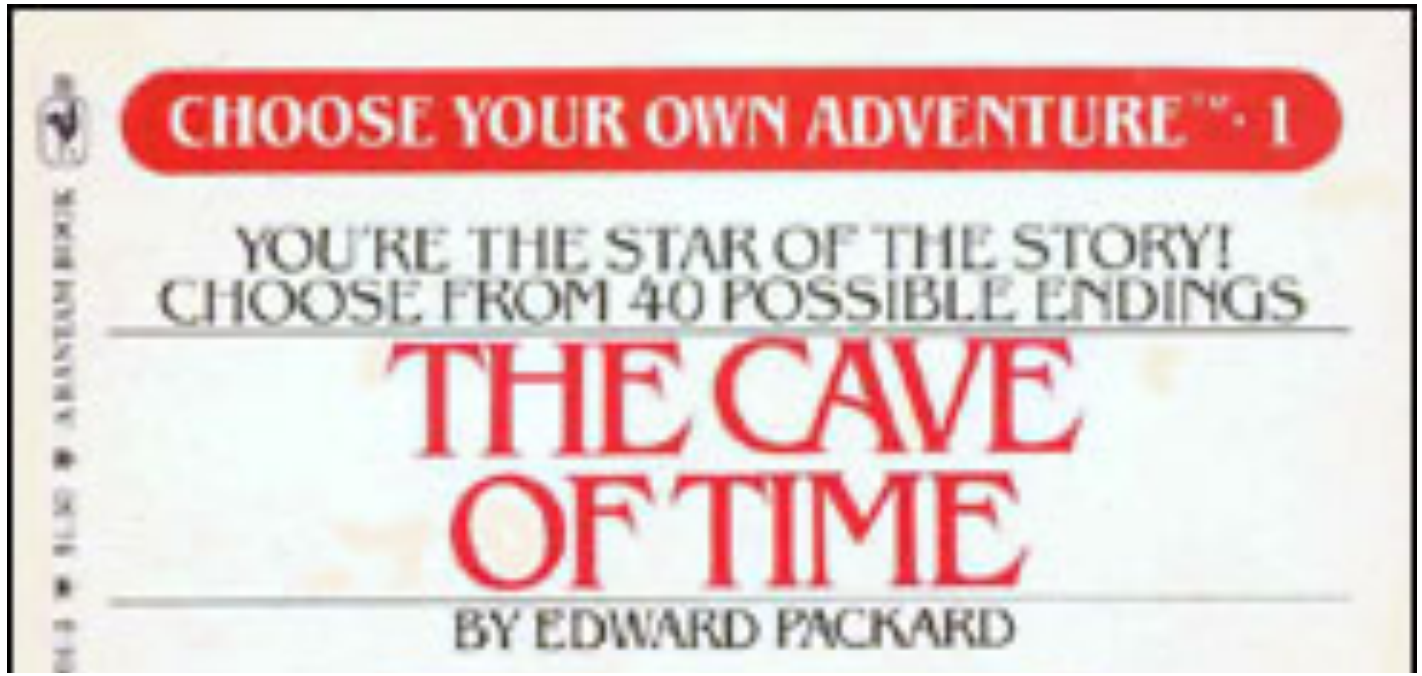
Operations reliability | Automate & react

- Mean time to recover
- Number of automated tickets
- Percentage of services automated
- Number of defects
- Percentage of proactive found vs. user found
- Cloud usage

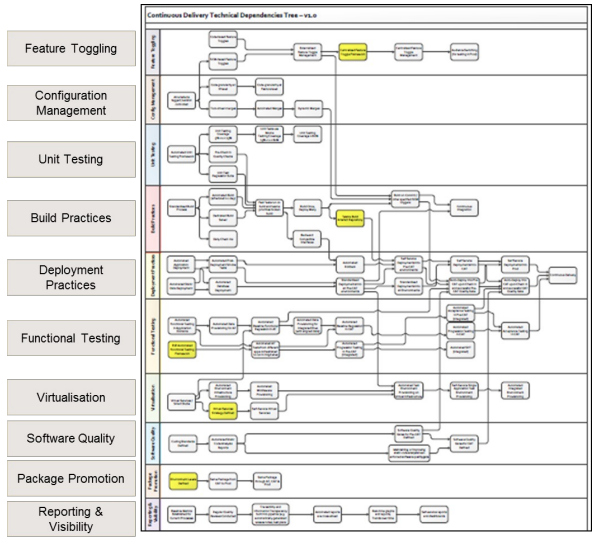
Architectural flexibility | Simplify & decouple

- Percentage of transactions in old vs. new
- Average size of release
- Percentage of interfaces that are backward compa
- Number of outdated components
- Number of security concerns
- Technical debt

Rule 3: Provide options



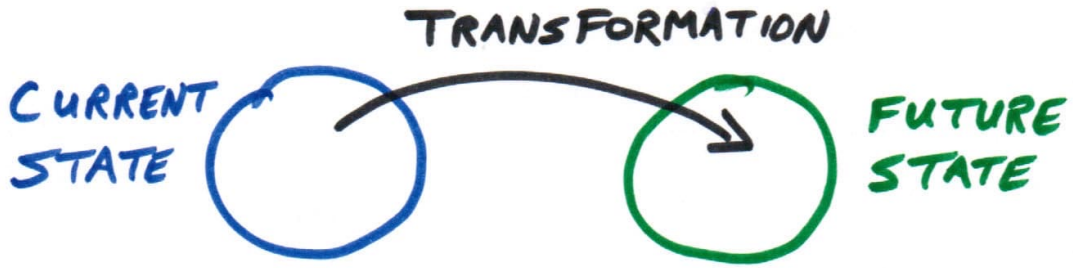
Rule 3: Provide guidance



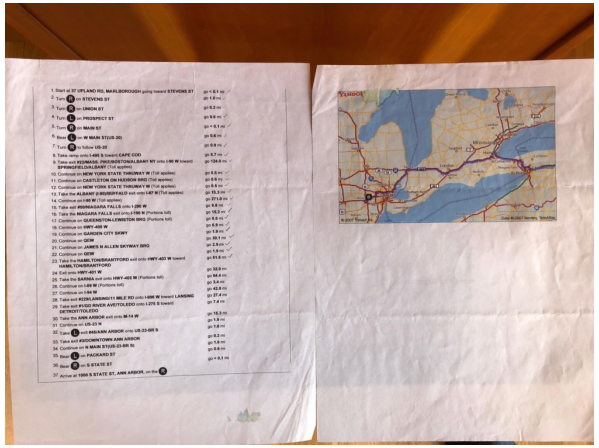
Enterprise DevOps Bingo Card

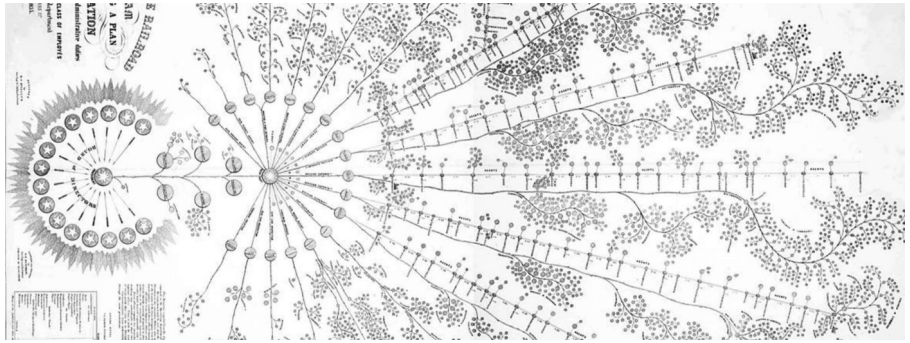
Governance	One Team	Test Automation & Coverage	Dev/Build Automation	Agile Delivery	Continuous Delivery	Lean Ops
Metrics ✓	Feature Teams ✓	Agile Test Strategy	CD pipeline ✓	Scrum Master ✓	Continuous Integration	Functional Monitoring
Dashboard	Owns Inception to Retirement	Unit Test Automation	DevSecOps	Agile Coach	Continuous Testing	Ticket Automation
Meeting Structure ✓	Success measured on App Fitness	End To End Test Auto. & coverage	Build/Deployment Automation	Backlog Health ✓	Continuous Deployment	App Insights and Ops Insights ✓
Contracting Approach ✓	Testing within Team ✓	Regression Test Automation	Infra as Code	BizDevOps ✓	End to End Release Automation	Fail Forward Deployments
Epic Management				Agile Documentation		Ops concerns in Scrum

Summary: Provide the teams with a compass, not a map



=





=

One step at a time… one conscious step at a time

Thank you