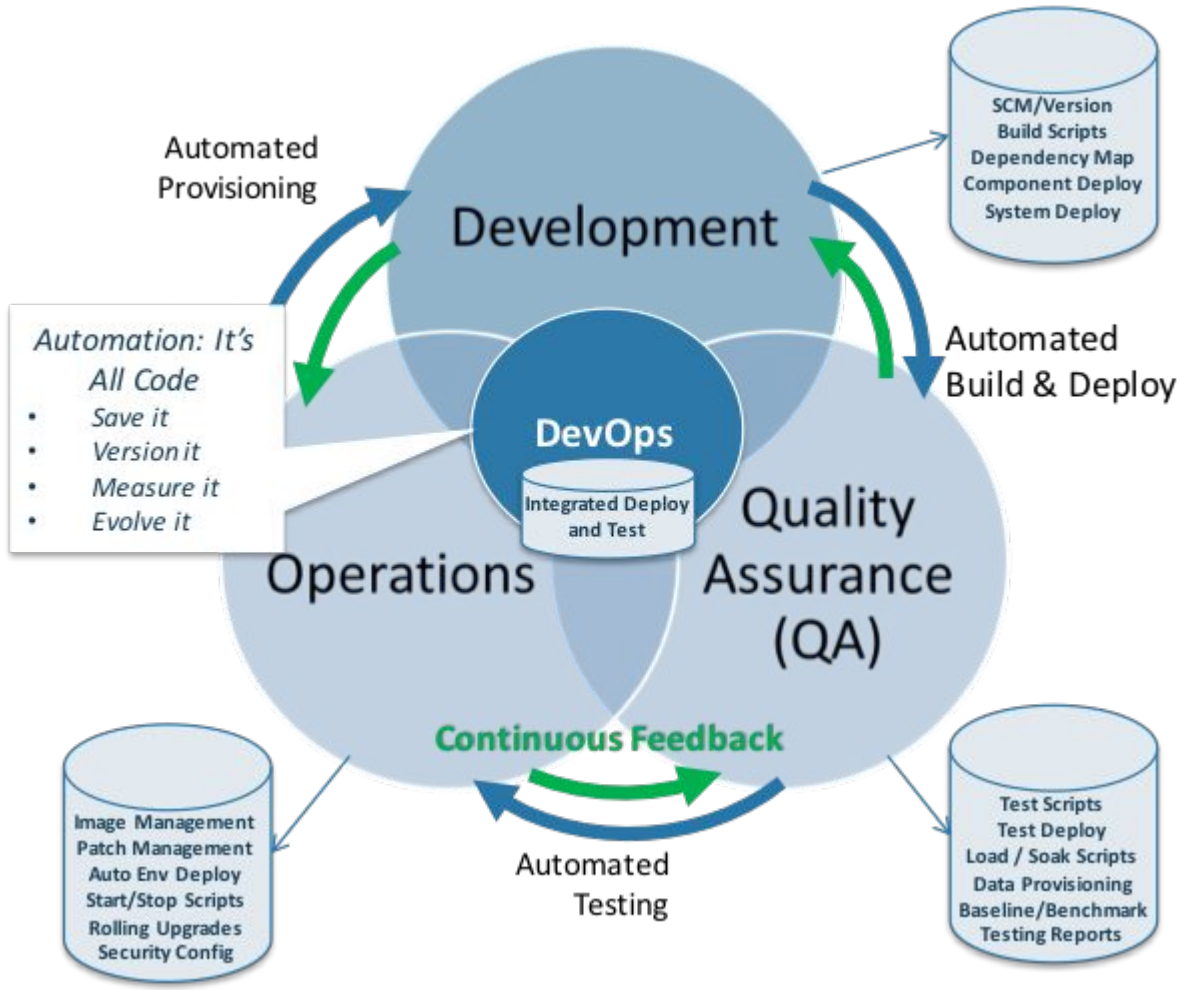
Rakamin

Soft Skill

Critical Thinking in IT

Rakamin


Automated
Provisioning
Development
SCM/Version
Build Scripts
Dependency Map
Component Deploy
System Deploy
Automated
Build & Deploy
Automation: It's
All Code
Save it
Version it
Measure it
Evolve it
DevOps
Integrated Deploy
and Test
Operations
Quality
Assurance
(QA)
Continuous Feedback
Automated
Testing
Image Management
Patch Management
Auto Env Deploy
Start/Stop Scripts
Rolling Upgrades
Security Config
Test Scripts
Test Deploy
Load / Soak Scripts
Data Provisioning
Baseline/Benchmark
Testing Reports

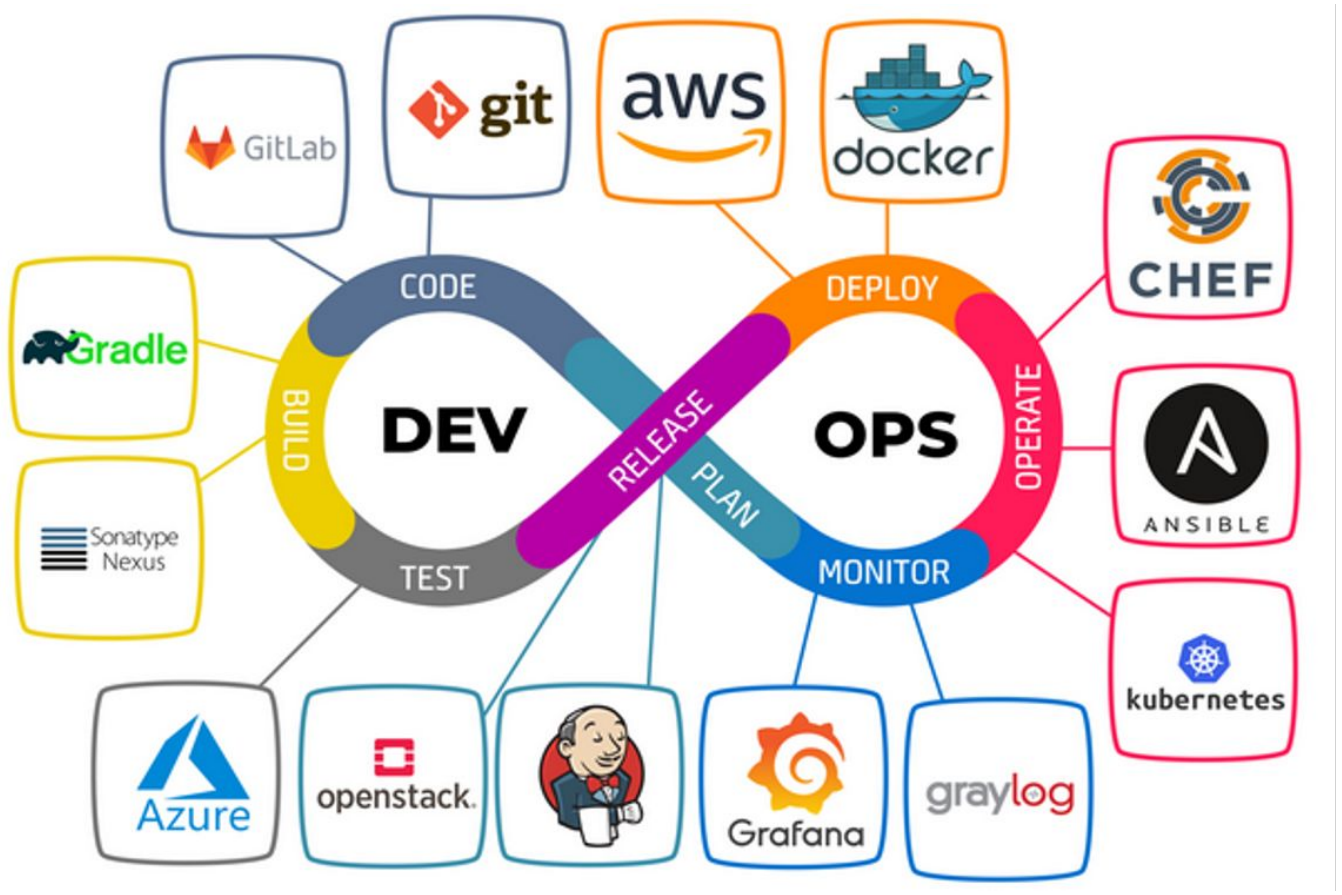
Rakamin

Plan | Code | Build | Test

Developer

Release | Deploy | Operate | Monitor

IT Operations



DevOps vs SRE vs Platform Engineer

Rakamin

DevOps:

Focus:
Bridging the **gap between development and operations** teams to improve collaboration and automate workflows.

Goals:
Speed up software development cycles, improve release quality, and reduce the time between releases.

Tools:
CI/CD pipelines, automation tools, monitoring and logging tools, infrastructure as code.

Responsibilities:
Managing CI/CD pipelines, deploying applications, monitoring infrastructure, and automating tasks.

Example:
A DevOps team might implement a CI/CD pipeline to automate the process of building, testing, and deploying an application.

Platform Engineering:

Focus:
Building and **managing internal platforms** and tools to streamline development and deployment, and **improve developer experience**.

Goals:
Increase developer productivity, provide a consistent and reliable development environment, and automate infrastructure provisioning.

Tools:
Infrastructure as code (IaC) tools, container orchestration platforms, and self-service APIs for infrastructure management.

Responsibilities:
Designing and implementing infrastructure, provisioning resources, automating processes, and providing tools for developers.

Example:
A platform engineering team might build an internal platform using Kubernetes to allow developers to request infrastructure resources.

SRE (Site Reliability Engineering):

Focus:
Ensuring the **reliability, performance, and availability** of applications and services.

Goals:
Minimize downtime, ensure system stability, and maintain service-level objectives (SLOs).

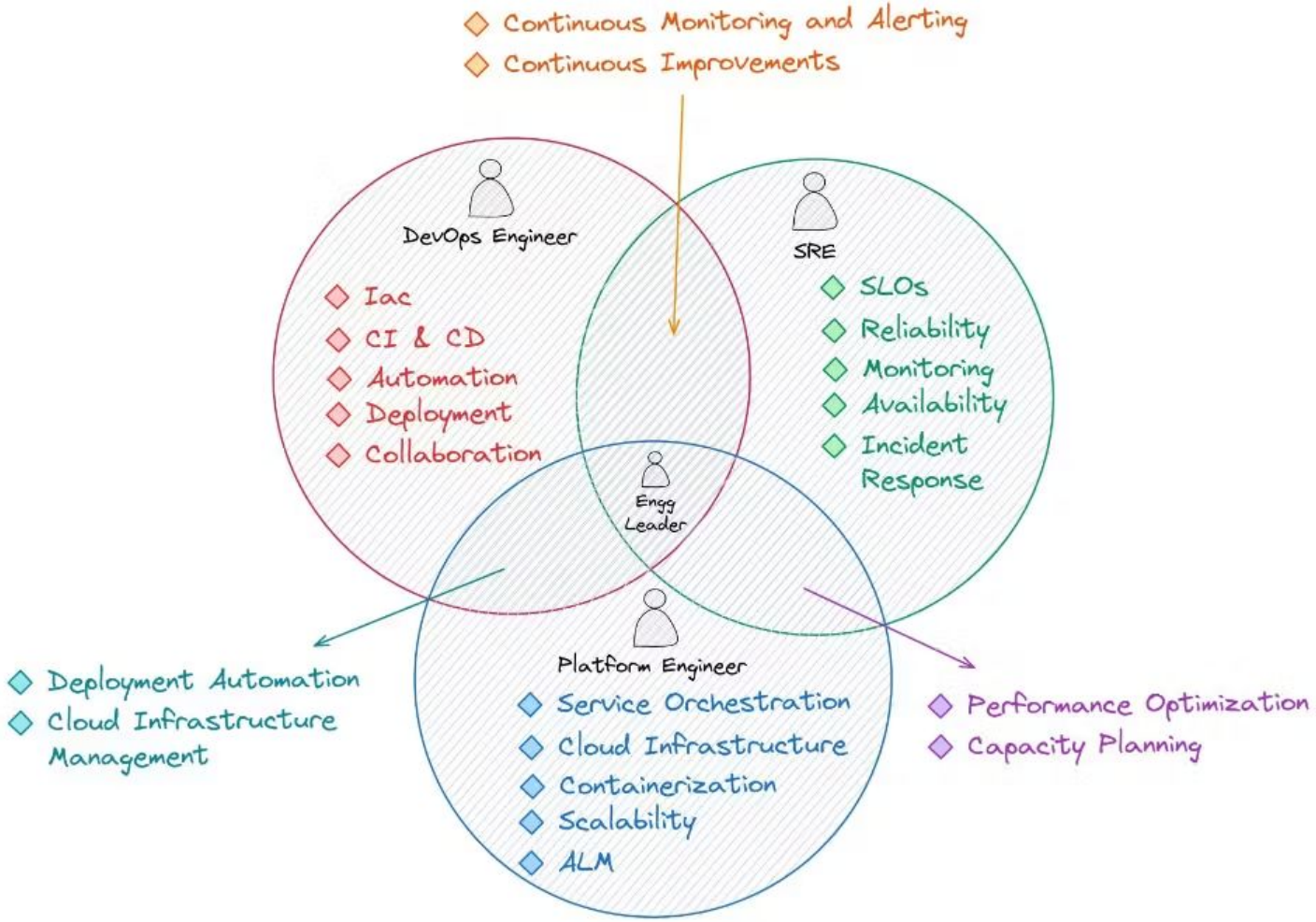
Tools:
Monitoring and alerting systems, chaos engineering tools, incident response systems.

Responsibilities:
Setting performance targets, monitoring infrastructure, detecting and resolving issues, and improving system stability.

Example:
An SRE team might set up monitoring with tools like Prometheus to detect failures and alerting to detect failures early, according to a blog post on Syntasso.

Rakamin

DevOps vs SRE vs Platform Engineer



https://www.splunk.com/en_us/blog/learn/sre-vs-devops-vs-platform-engineering.html

Mengembangkan Pemikiran Kritis dalam IT

Kemampuan untuk menganalisis informasi secara objektif, mengevaluasi bukti, dan mengambil keputusan logis untuk menyelesaikan masalah teknis.

Strategi Pengembangan:

1. **Latih Pertanyaan Kritis**:
   - Contoh:
     - "Apa bukti bahwa server down disebabkan oleh overload, bukan bug kode?"
     - "Mengapa solusi X lebih efektif daripada Y untuk masalah ini?"
2. **Belajar dari Kegagalan**:
   - Analisis post-mortem insiden (contoh: outage Tokopedia 2021).
   - Tools: **Root Cause Analysis (RCA)** atau **5 Whys**.
3. **Eksplorasi Multi-Disiplin**:
   - Integrasi pengetahuan jaringan, programming, dan keamanan siber.
   - Contoh: Memahami bagaimana bug di frontend (React) bisa memicu kerentanan backend (SQLi).
4. **Gunakan Framework Terstruktur**:
   - **OODA Loop** (Observe, Orient, Decide, Act).
   - **ITIL Problem Management**.

Rakamin

Mengidentifikasi Masalah Teknis Kompleks

Langkah Sistematis:

1. **Definisikan Masalah dengan Jelas**:
   - Contoh:
     - *Salah*: "Aplikasi lambat."
     - *Benar*: "Response time API pembayaran meningkat 300% saat peak hour."
2. **Kumpulkan Data Relevan**:
   - Log server, metrik CPU/RAM, tracing (OpenTelemetry).
   - Tools: **Prometheus**, **Grafana**, atau **ELK Stack**.
3. **Break Down Masalah**:
   - Contoh:
     - *Masalah*: Error 500 pada endpoint `/checkout`.
     - *Breakdown*:
       - Database timeout?
       - Microservice payment gateway gagal?
       - Kesalahan konfigurasi load balancer?
4. **Validasi Hipotesis**:
   - Contoh:
     - Hipotesis: "Database timeout karena query tidak optimal."
     - Validasi: Cek slow query log & analisis dengan `EXPLAIN` di PostgreSQL.

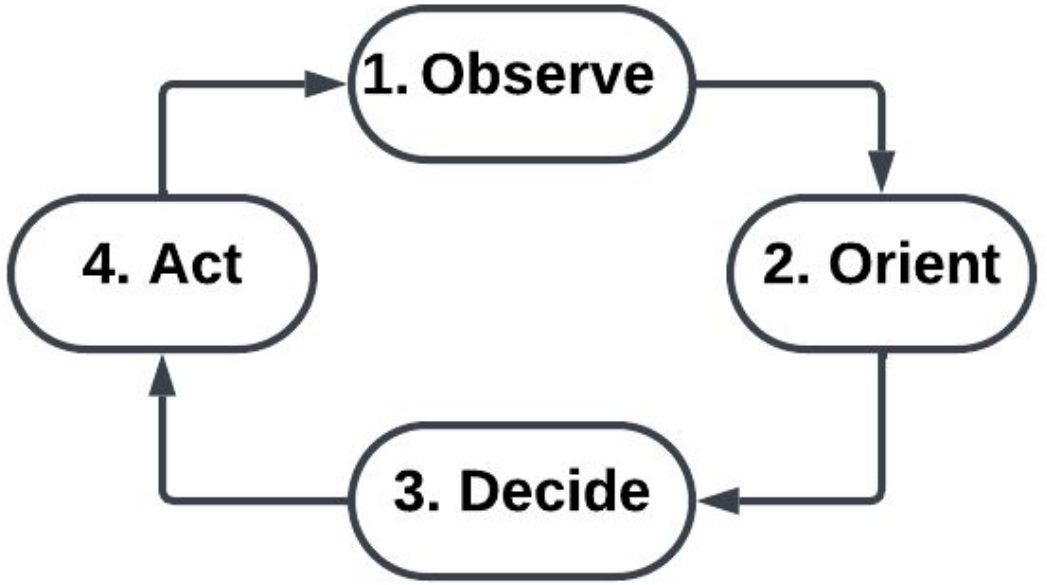
Rakamin

Hands-On: Simulasi Analisis Root Cause

Kasus: Aplikasi e-commerce tiba-tiba mengalami penurunan traffic 70% setelah update.

Langkah Analisis:

1. **Observe (Pengamatan)**:
   - Gejala:
     - Halaman produk gagal load (error 502).
     - Server database menunjukkan penggunaan CP 95%.
   - Data:
     - Log error: Too many connections to MySQL.
     - Metrics: Spike request ke endpoint /product-de
2. **Orient (Identifikasi Pola)**:
   - Timeline: Masalah mulai setelah deploy fitur "rekomendasi produk real-time".
   - Bandingkan dengan versi sebelumnya:
     - Kode baru menggunakan query SQL tanpa indeks.
     - Konfigurasi connection pool database tidak diupdate.



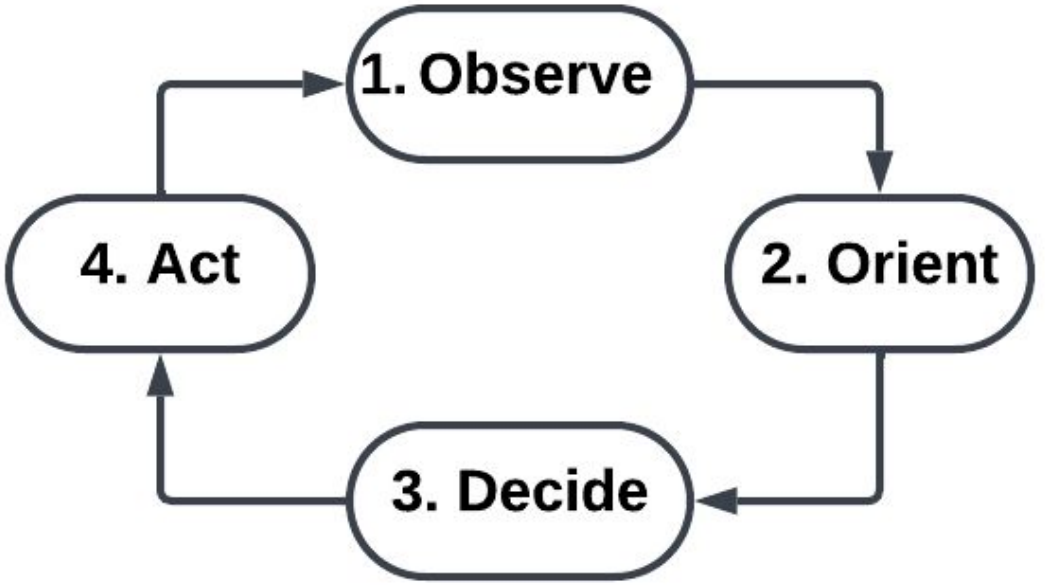
Rakamin

Hands-On: Simulasi Analisis Root Cause

Kasus: Aplikasi e-commerce tiba-tiba mengalami penurunan traffic 70% setelah update.

Langkah Analisis:

3. **Decide (Keputusan Sementara)**:
   - Hipotesis:
     - Query pada fitur rekomendasi menyebabkan bottleneck.
     - Connection pool overflow karena thread meningkat.
4. **Act (Validasi & Solusi)**:
   - Validasi:
     - Jalankan `SHOW PROCESSLIST` di MySQL → Banyak query `SELECT * FROM products WHERE ...` tanpa indeks.
     - Test load dengan JMeter: Error terjadi saat concurrent user > 500.

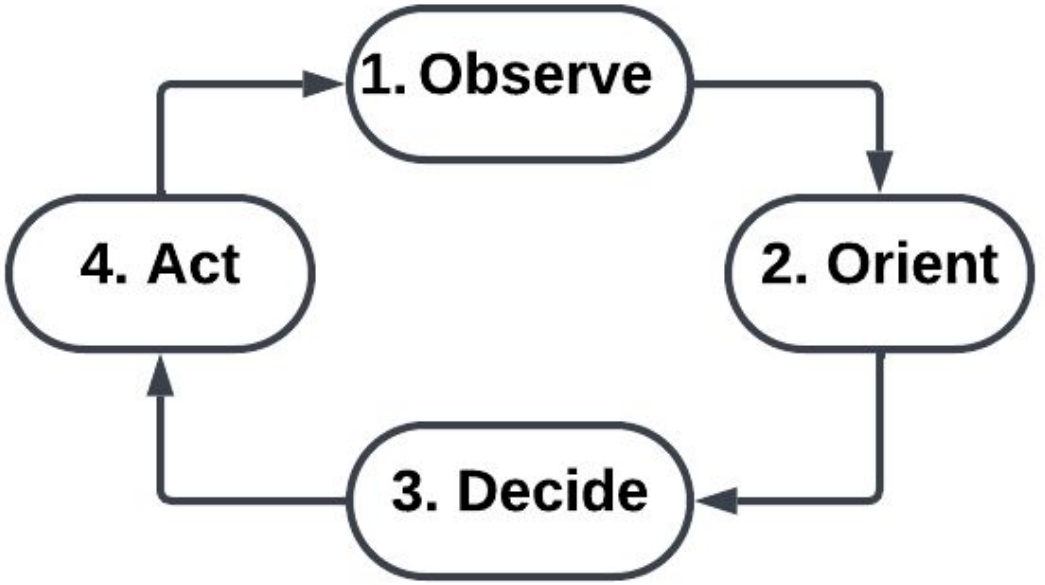


Rakamin

Hands-On: Simulasi Analisis Root Cause

Solusi:

- Tambahkan indeks pada kolom `category_id`.
- Optimalkan connection pool (max_connection dari 100 → 300).
- Rollback sementara fitur rekomendasi.



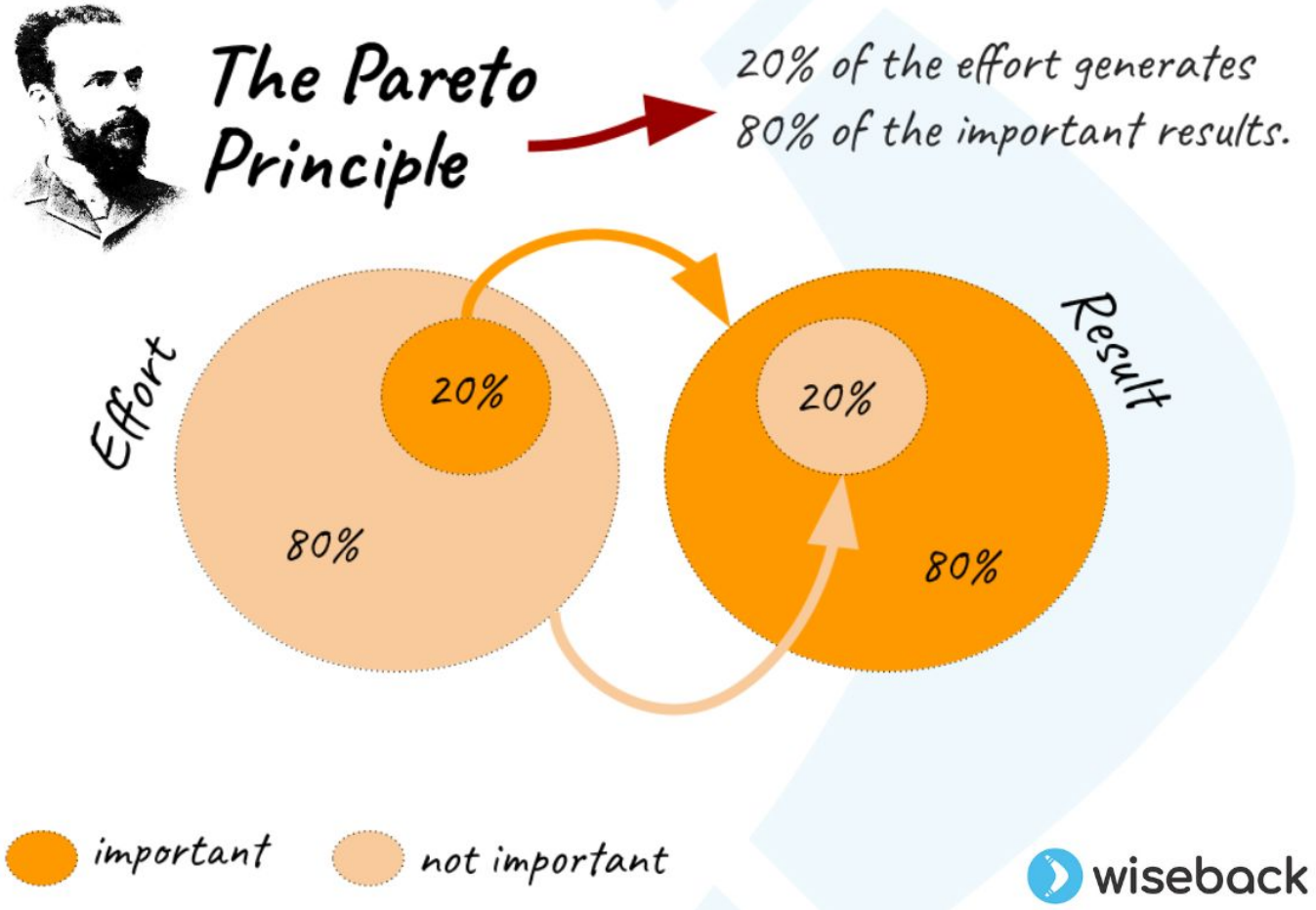
Rakamin

Kesalahan Umum yang Harus Dihindari

1. **Confirmation Bias**: Hanya mencari bukti yang mendukung asumsi awal.
    - Contoh: Menyalahkan infrastruktur tanpa memeriksa kode.
2. **Over-Simplifikasi**: Menganggap masalah kompleks bisa diselesaikan dengan solusi instan.
3. **Mengabaikan Dokumentasi**: Tidak mereplikasi masalah karena kurang data historis.

Rakamin

Framework: First Principle Thinking



Rakamin

Framework: Pareto Principle Thinking

First Principle Canvas

Find problem the what stage	**Clarify problem** the how stage
Find the top three most important and/or urgent problems.	Gain as much insights of the problem as possible. What exactly is the problem and how did it come to be.
Break down problem the why stage	**Solve problem**
Breakdown the problem to the most fundamental components.	Use the previous stages to come with a solution that the company can carry.
Question the process playing devils advocate	
Question the entire process. Reiterate when needed.	

References

https://x.com/ibamarief/status/1744672991072682318?t=5LFfrgvb2qmWgc_FWGwCpQ&s=08

Gojek :
https://www.gojek.io/blog/why-we-swear-by-the-rca

Uber:
https://assets.nextleap.app/submissions/RootCauseAnalysisRCA_UbersDeclineinDAUs-bc7e8279-0564-4ead-bf17-39ab5be9efe6.pdf

Tokopedia:
https://katadata.co.id/digital/e-commerce/5febfcfee3674/engineer-tokopedia-berbagi-cerita-menangani-dan-mengelola-insiden

https://medium.com/tokopedia-engineering/embracing-on-call-system-in-software-development-process-c74b030e6565

Public Incident Reports dari Perusahaan Besar

A. Google

- **Contoh Post-Mortem**:
  - Google Cloud Incident #20013 (2023)
  - **Apa yang Dipelajari**: Gangguan global karena konfigurasi firewall yang salah.
- **Blog Resmi**: Google Cloud Blog - Incident Management

B. Amazon Web Services (AWS)

- **AWS Service Health Dashboard**
  - Contoh: AWS US-EAST-1 Outage (2021)
  - **Root Cause**: Kesalahan automasi jaringan yang memicu cascade failure.

C. Microsoft Azure

- **Azure Status History**
  - Contoh: Azure AD Outage (2022)
  - **Pelajaran**: Bug pada token caching menyebabkan autentikasi global gagal.

D. Slack

- **Slack Post-Mortem Archives**

https://github.com/danluu/post-mortems https://github.com/decentraland/rca?tab=readme-ov-file

Rakamin

TERIMA KASIH