Journey in Adopting Cloud-Native Technology

Aditya Satrya

Head of IT Development at Jabar Digital Service

Meetup Kubernetes/
Cloud-Native Bandung #4

Context

Jabar Digital Service

- Data-driven policy making
- Digital transformation
- Innovation in public service
- Build digital ecosystem to give benefit to citizen

We are hiring engineers in early 2020

Stay tune =)

What we build

- In-house
  - Sapawarga (our main focus)
  - Citizen single-sign on
  - Crowdsourcing App
  - Many dashboards
- With vendors
  - 12 applications

Agility (Speed & Quality) Metrics

No	Name	Detail
1	Deployment Frequency	API: 14x deploy to production Web Admin 17x deploy to production Mobile Ionic: 3x deploy to beta
2	Uptime	99.99%
3	Bug Report External & Internal	Eksternal: 0 Internal: 8(3 closed, 5 remaining)


Selamat Datang Peserta Pelatihan
Aplikasi
sapawarga
"Dari Kami, Untuk Masyarakat Jawa Barat"
sapawarga


BIMBINGAN TEKNIS
APLIKASI SAPAWARGA


sapawarga

sapawarga


SELAMAT DATANG
DI KANTOR KEPALA DESA MARGASARI


SERANGMEKAR


Selamat Datang Peserta Pelatihan
Aplikasi
sapawarga

Cloud-Native Definition

Cloud-native is about how applications are created and deployed, not where — Pivotal

Cloud-native is an approach to building and running applications that exploits the advantages of the cloud computing
— Pivotal

Cloud-native architecture means adapting to the many new possibilities offered by the cloud compared to traditional on-premises infrastructure.
— Google Cloud Blog

One of the core characteristics of a cloud-native system is that it’s always evolving — Google Cloud Blog

Cloud-native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds — CNCF

These techniques enable loosely coupled systems that are resilient, manageable, and observable — CNCF

Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil — CNCF

Cloud-Native is...

- A way in building and running applications
- Exploit cloud computing advantages

What advantages?

- Scalable
- Loosely coupled
- Resilient
- Manageable
- Observable
- Frequent changes / speed

Start with the right mindset

Werner Vogels

@Werner

Follow

in one tweet? 1) if you want to be a VP of Engineering focus on teams and people and how to make them succesfull 2) wanna be a CTO? Simplify. Focus on the business, what is the simplest, most robust Tech/Ops that makes the business succeed.

Dennis @PilatDennis

@Werner Hi Werner, I was wondering... what advice would you offer to somebody who'd like to be a CTO someday? Especially for the younger folks, what should one know or how should one prepare themselves for their future?…

As an engineer / CTO,
try to constantly asks these questions...

Do you have a clear understanding on what business success really is?

Is this the simplest solution to start with?

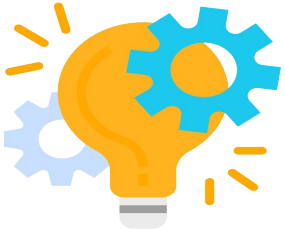
What investment in tooling should we spend to make engineers more productive?

Build the right culture

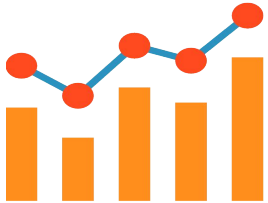
Set your organization values

RESPONSIF

INOVATIF

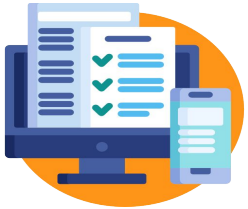


ADAPTIF

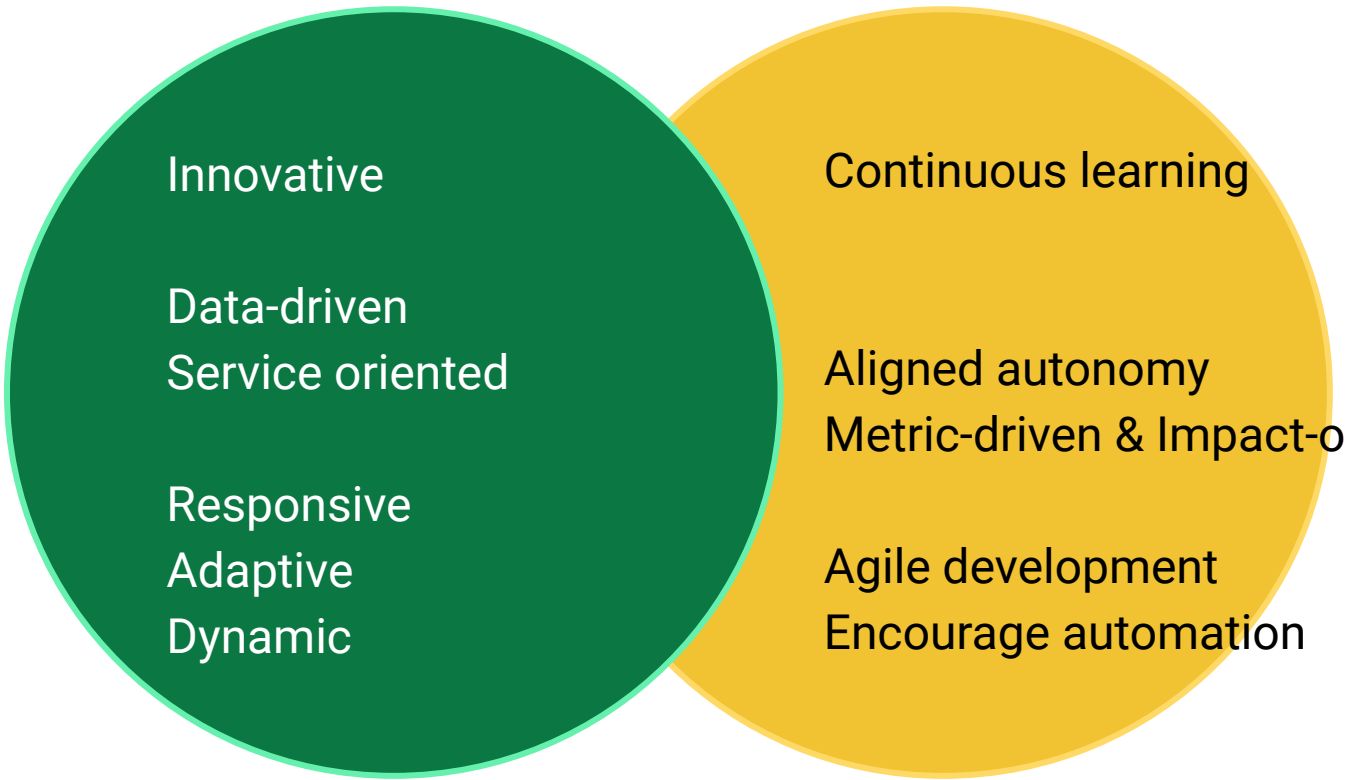


DATA DRIVEN

SERVICE ORIENTED



DYNAMIC


Organization Values
Innovative
Data-driven
Service oriented
Responsive
Adaptive
Dynamic
Engineering Culture
Continuous learning
Aligned autonomy
Metric-driven & Impact-oriented
Agile development
Encourage automation
Engineering Practice
TechTalk
360° Feedback
OKR
Scrum
CI/CD

Continuous learning

Open to changes

It’s safe to make a mistake

Right balance: learning curve & delivering business values

Aligned autonomy

Decentralized decision making

Product ownership

Metric-driven & Impact-oriented

Experimentation mindset

Set metrics (key results)

Agile

Welcoming changes

Deliver business value as early as possible

Encourage Automation

Be lazy

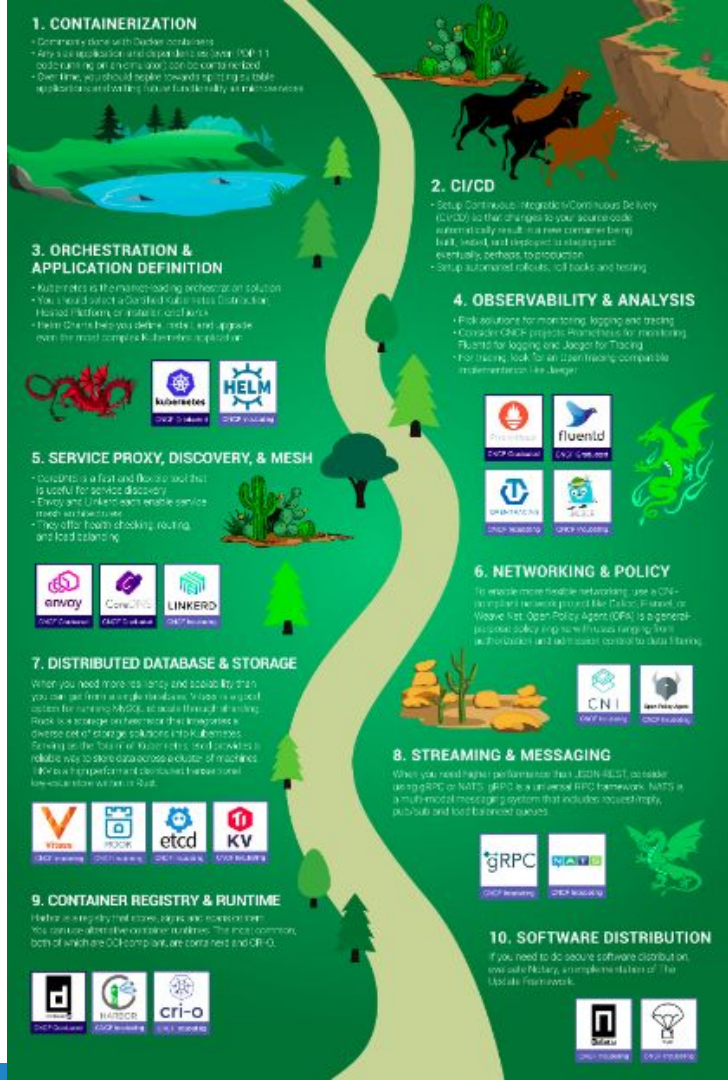
Follow Trail Map &
Always Take Lessons Learned

Cloud-Native Trail-map

1. Containerization
2. CI/CD
3. Orchestration

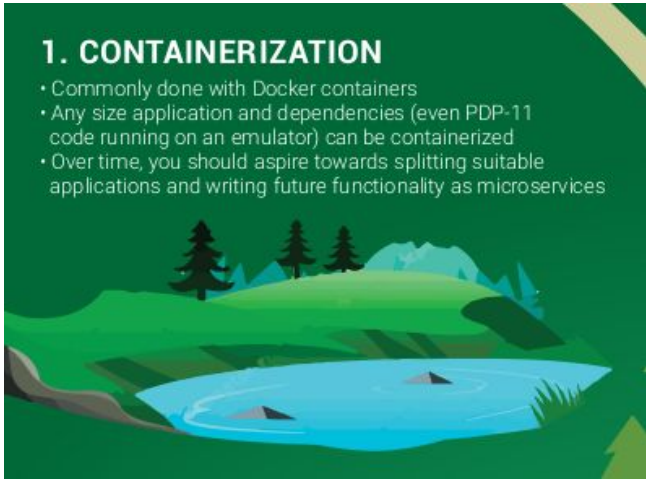
--below this are optional--

4. Observability
5. Service Discovery
6. Networking & Policy
7. Distributed database & storage
8. Streaming & messaging
9. Container registry
10. Software distribution



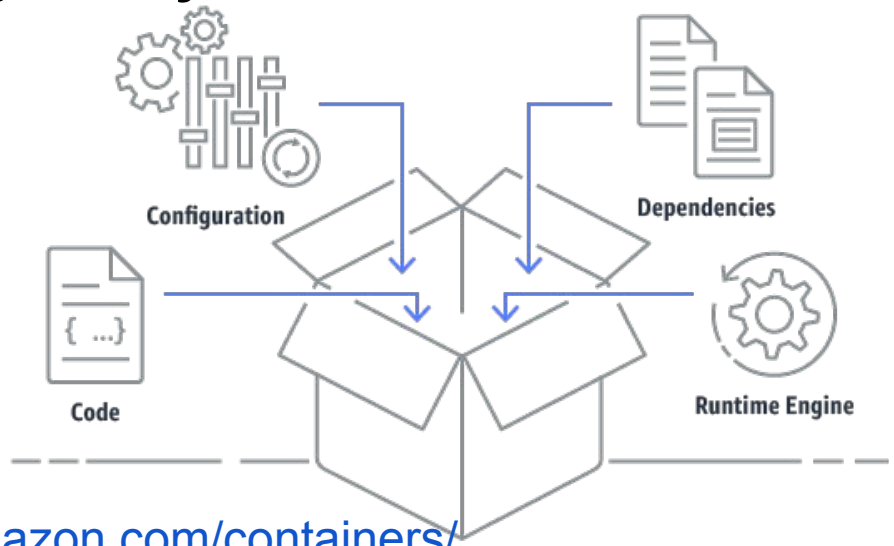
Step 1: Containerization

- You can't do cloud-native without containerization
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices



Containerization

- a standard way to package your application's code, configurations, and dependencies into a single object.



https://aws.amazon.com/containers/

Benefit

- Run anywhere
- Scale quickly
- Better resource utilization

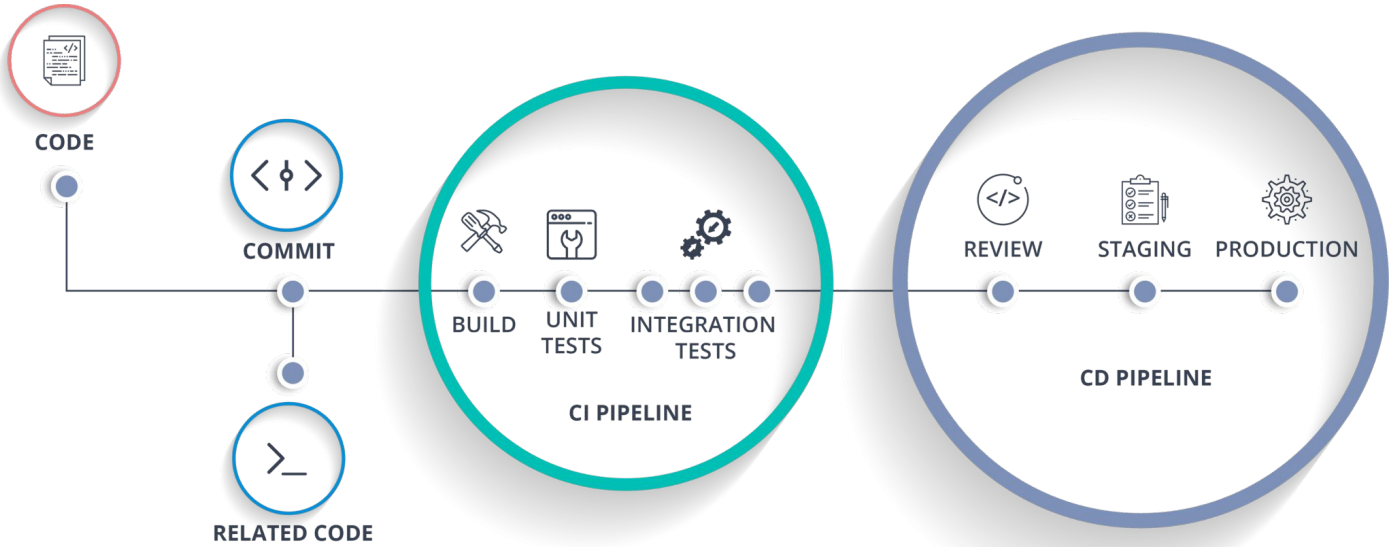
Lessons Learned

- Worth learning curve to invest
- Faster onboarding time for new devs
- Start with monolith
- Start simple: run containers in server using docker compose

Step 2: CI/CD

- Changes to your source code automatically result in a new container being built, tested, and deployed to staging and (eventually) to production
- Automated rollouts & rollbacks

CI/CD



Source: https://dzone.com/articles/learn-how-to-setup-a-cicd-pipeline-from-scratch

Komponen	Coding	Continuous Integration	Continuous Delivery	Continuous Deployment	Continuous Monitoring
API	Git branching	Code review	Automatic versioning	Automatic deploy to staging	Centralized logging
	API mocking	Automated unit testing	Automatic build	Automatic deploy to production	Infrastructure logging & monitoring
	Maintainability checking	Automated functional testing	Automatic release		Request logging & monitoring
	Code coverage checking	Automated end-to-end testing			DB transaction logging & monitoring
	Code linting	Security testing			Code-level performance analytics
					Error logging
Webadmin	Git branching	Code review	Automatic versioning	Automatic deploy to staging	Centralized logging
	API mocking	Automated unit testing	Automatic build	Automatic deploy to production	Infrastructure logging & monitoring
	Maintainability checking	Automated functional testing	Automatic release		Request logging & monitoring
	Code coverage checking*	Automated end-to-end testing			Code-level performance analytics
	Code linting	Security testing			Error logging
Mobile Ionic	Git branching	Code review	Automatic versioning	Automatic deploy (for internal)	Centralized logging
	API mocking	Automated unit testing	Automatic build		Usage logging & monitoring
	Maintainability checking	Automated functional testing	Automatic release		Event logging & monitoring
	Code coverage checking	Automated end-to-end testing			Crash logging & monitoring
	Code linting	Security testing			Triggered alerting
					Code-level performance analytics
Mobile Flutter	Git branching	Code review	Automatic versioning	Automatic deploy (for internal)*	Centralized logging
	API mocking	Automated unit testing	Automatic build		Usage logging & monitoring
	Maintainability checking	Automated functional testing	Automatic release		Event logging & monitoring
	Code coverage checking*	Automated end-to-end testing			Crash logging & monitoring
	Code linting	Security testing			Triggered alerting
					Code-level performance analytics

Done

In Progress

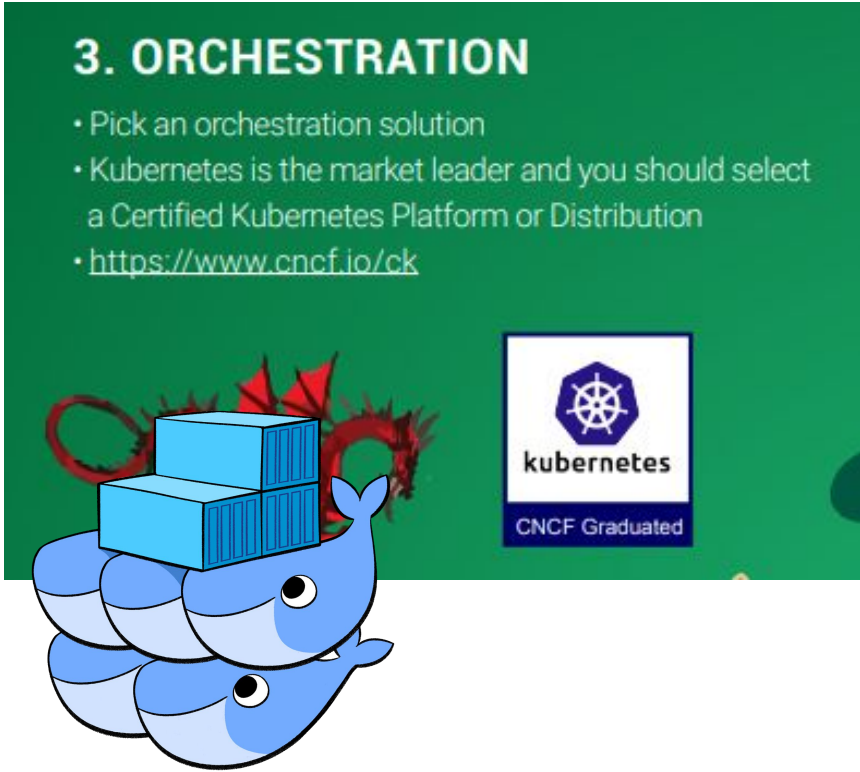
Not Yet

Lessons Learned

- Implement CI/CD from the beginning
- Start simple: just build and deploy
- Maintain checklists and set goals
- You need at least 1 developer who understand this better

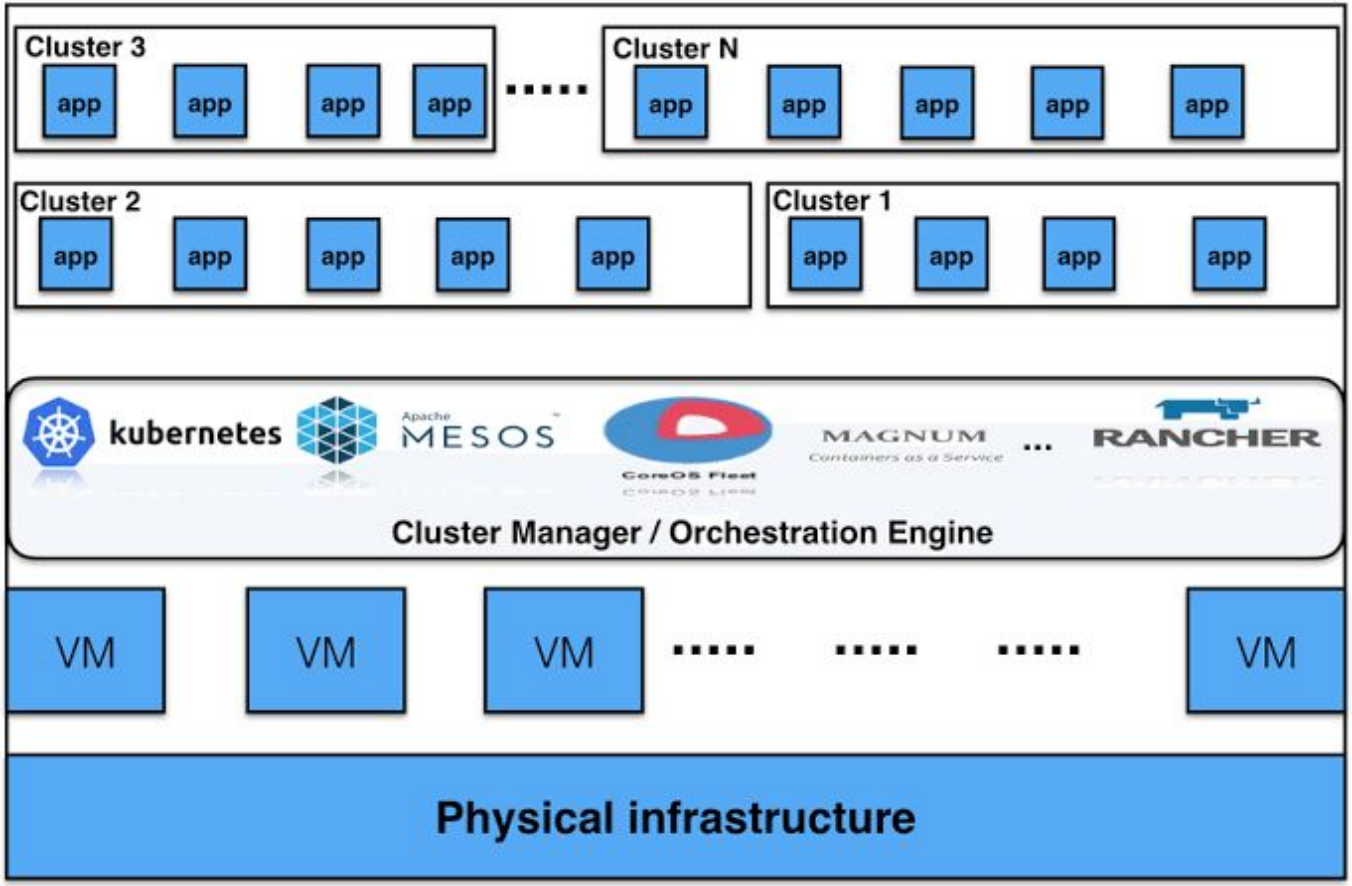
Step 3: Orchestration

- Pick an orchestration solution
- Kubernetes is the market leader



Automate Container Management

- Configuration
- Provisioning
- Availability
- Scaling
- Security
- Resource allocation
- Load balancing



Source: https://www.researchgate.net/figure/Container-orchestration-engines_fig6_317297877

Lessons Learned

- Kubernetes works for monolith
- Don’t use Kubernetes as your first production solution
  - You should have another working solution before Kubernetes
- Use managed kubernetes service in the first place
- Be mindful about the cost (money)
- Don’t use Kubernetes before CI/CD
- Kubernetes give you a solid foundation to scale and implement many best practices

Summary

- Adopting cloud-native is not all-technical aspect
  - Mindset
  - Culture
  - Technical
- It’s not only IT Division’s matter, but entire organization