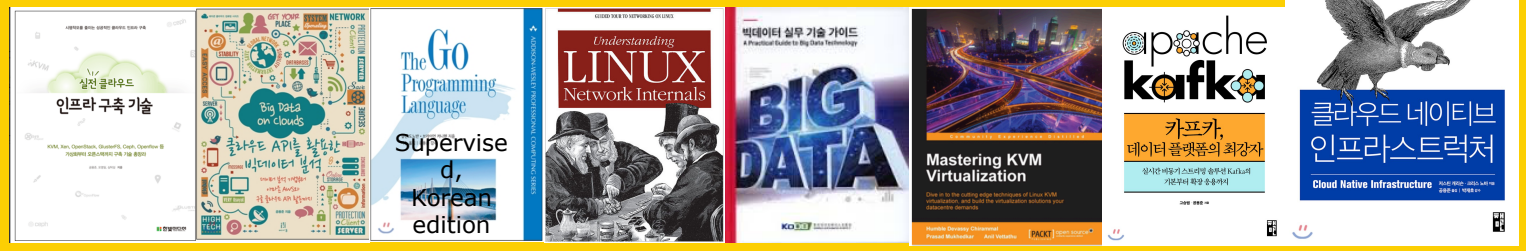
To the Zero Operation with cloud

공용준(Andrew.kong)
kakao corp

Who am I

Andrew. Yongjoon kong

- Cloud Technical Advisory for Government Broad Cast Agency
- Adjunct Prof. Ajou Univ
- Korea Data Base Agency Acting Professor for Bigdata
- Member of National Information Agency Bigdata Advisory committee
- Kakaocorp, Cloud Tech Director
- Talks
  - Scalable Loadbalancer with VM orchestrator (2017, netdev, korea)
  - Embrace clouds (2017, openstack days, korea)
  - Full route based network with linux (2016, netdev, Tokyo)
  - SDN without SDN (2015, openstack, Vancouber)



First of All

- What is Cloud?



if (kakao) : Cloud.define()

Programmable **Resource** Management	
2016	What is Resource?



Programmable Resource **Life Cycle** Management	
2017	What is Life Cycle?



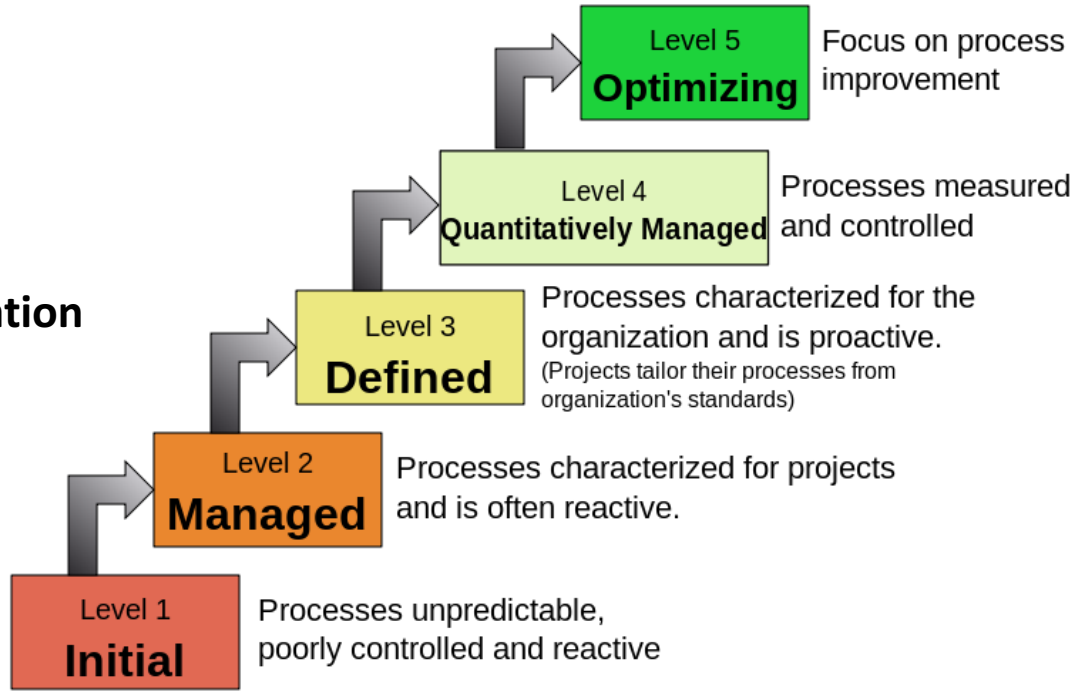
Programmable **Service** Management	
2018	What is Service?

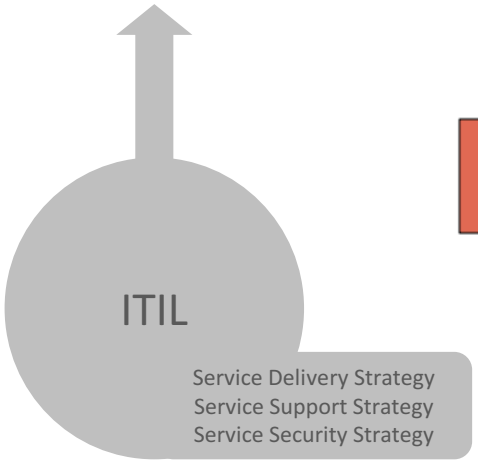
How Far can you go with your cloud?

CMMI Model

Capability Maturity Model Integration

developed by CMU



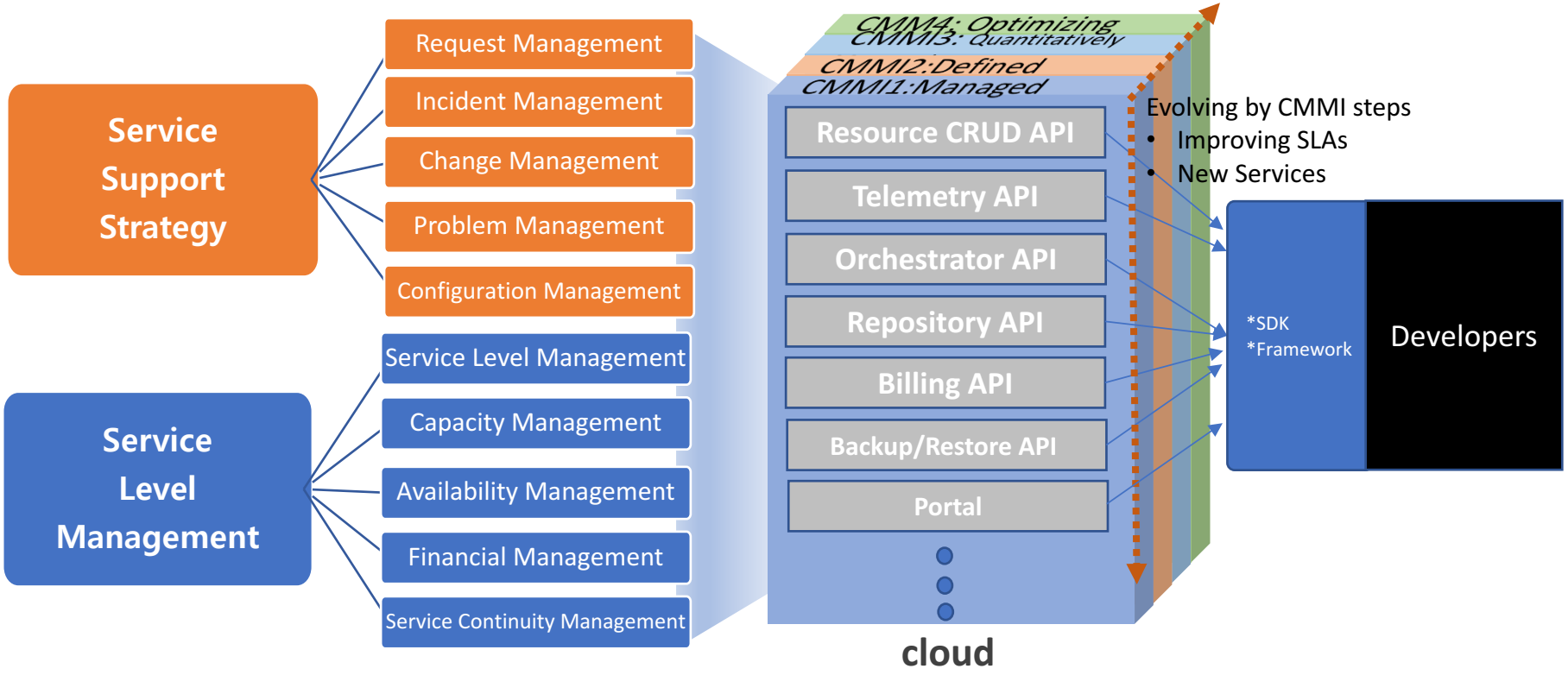


Now it’s time to ask

What’s for?

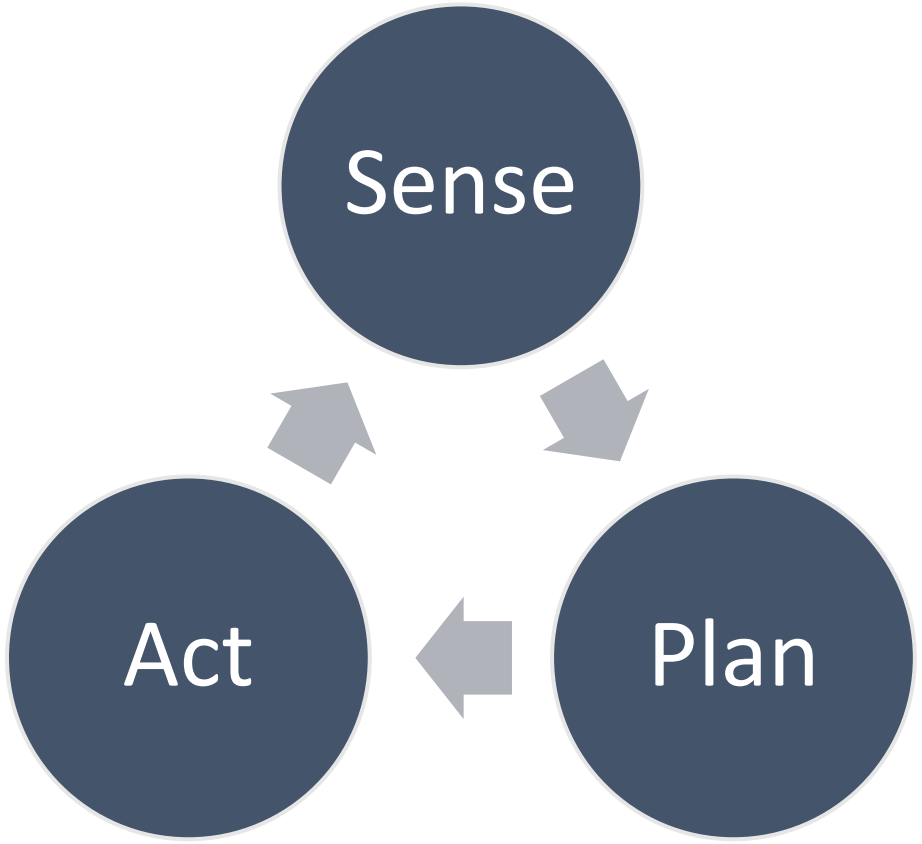


The overall IT strategy with cloud



The base of ITIL is:

Continual Improvement



CMMI Dev perspective

Maturity → Targets

	Initial	Managed	Defined	Quantitatively Managed	Optimizing
Culture & Organization	• Teams organized based on platform/ technology • Defined and documented processes	• One backlog per team • Adopt agile methodologies • Remove team boundaries	• Extended team collaboration • Remove boundary dev/ ops • Common process for all changes	• Cross-team continuous improvement • Teams responsible all the way to production	• Cross functional teams
Build & Deploy	• Centralized version control • Automated build scripts • No management of artifacts • Manual deployment • Environments are manually provisioned	• Polling CI builds • Any build can be re-created from source control • Management of build artifacts • Automated deployment scripts • Automated provisioning of environments	• Commit hook CI builds • Build fails if quality is not met (code analysis, performance, etc.) • Push button deployment and release of any releasable artifact to any environment • Standard deployment process for all environments	• Team priorities keeping codebase deployable over doing new work • Builds are not left broken • Orchestrated deployments • Blue Green Deployments	• Zero touch Continuous Deployments
Data Management	• Data migrations are performed manually, no scripts	• Data migrations using versioned scripts, performed manually	• Automated and versioned changes to datastores	• Changes to datastores automatically performed as part of the deployment process	• Automatic datastore changes and rollbacks tested with every deployment
Test & Verification	• Automated unit tests • Separate test environment	• Automatic Integration Tests • Static code analysis • Test coverage analysis	• Automatic functional tests • Manual performance/ security tests	• Fully automatic acceptance tests • Automatic performance/security tests • Manual exploratory testing based on risk based testing analysis	• Verify expected business value • Defects found and fixed immediately (roll forward)
Information & Reporting	• Baseline process metrics • Manual reporting • Visible to report runner	• Measure the process • Automatic reporting • Visible to team	• Automatic generation of release notes • Pipeline traceability • Reporting history • Visible to cross-silo	• Report trend analysis • Real time graphs on deployment pipeline metrics	• Dynamic self-service of information • Customizable dashboards • Cross-reference across organizational boundaries

CMM1: Initial

- Centralized Version Control
- Automated build script
  - No management of Artifacts
- Manual deployment
- Manual Environment provision

CMM2: Managed

- Polling CI builds
- Any build can be re-created from source control
- Management of build artifacts
- Automated deployment scripts

CMM3: Defined

- Commit hook CI builds
- Build fails if quality is not met (code analysis …)
- Push button deployment and release of any releasable artifacts to any environment
- Standard deployment process for all environments

CMM4: Quantitatively managed

- Team priorities keeping code base deployable over doing new work
- Builds are not left broken
- Orchestrated deployments
- Blue Green Deployments

CMM5: Optimized

- Zero touch continuously deployment
- The thing is that you need time, or

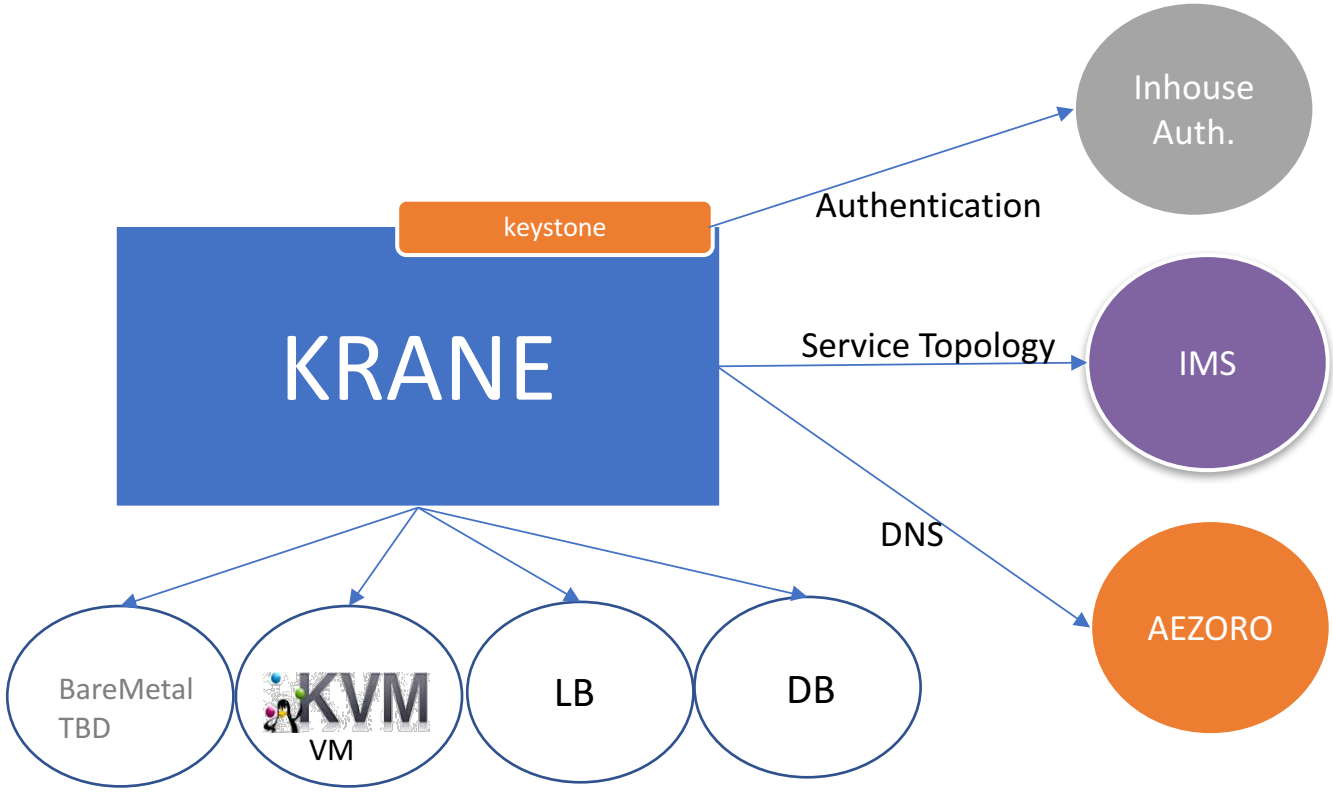


if (kakao): IaaS

What is the purpose of doing IaaS?

CMM0	CMM1
legacy	self service Dev resource
output: ITF	output: krane (openstack cloud)

if (kakao): IaaS (service name KRANE)

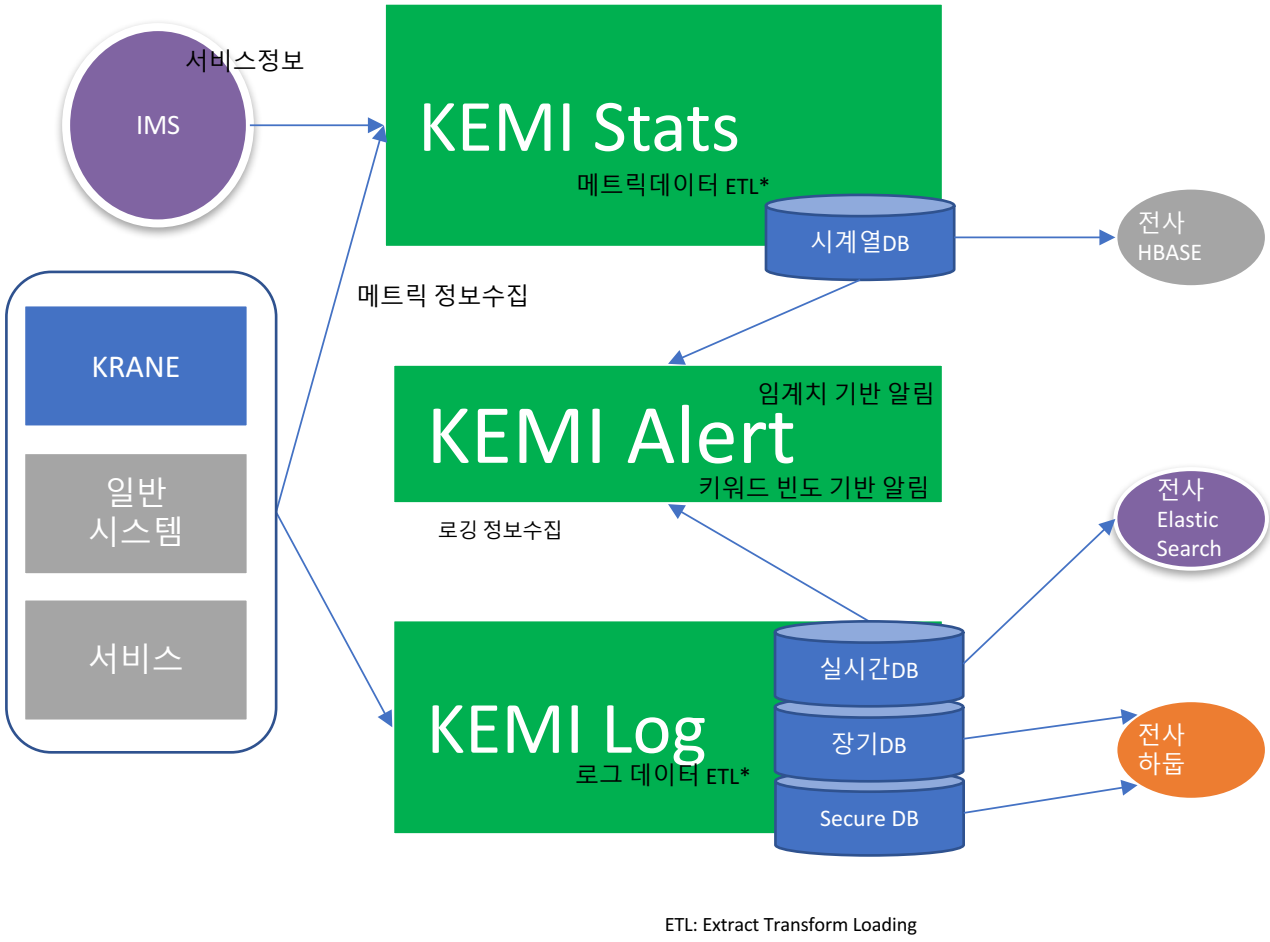


Cloud CMMI2-MaaS

What is the sole purpose of doing MaaS?

CMM0	CMM1	CMM2
legacy	self service Dev resource	limited Prod resources
output: cloudTF	output: krane (openstack cloud)	output: kemi (MaaS)

if (kakao): MaaS (service name KEMI)



ETL: Extract Transform Loading

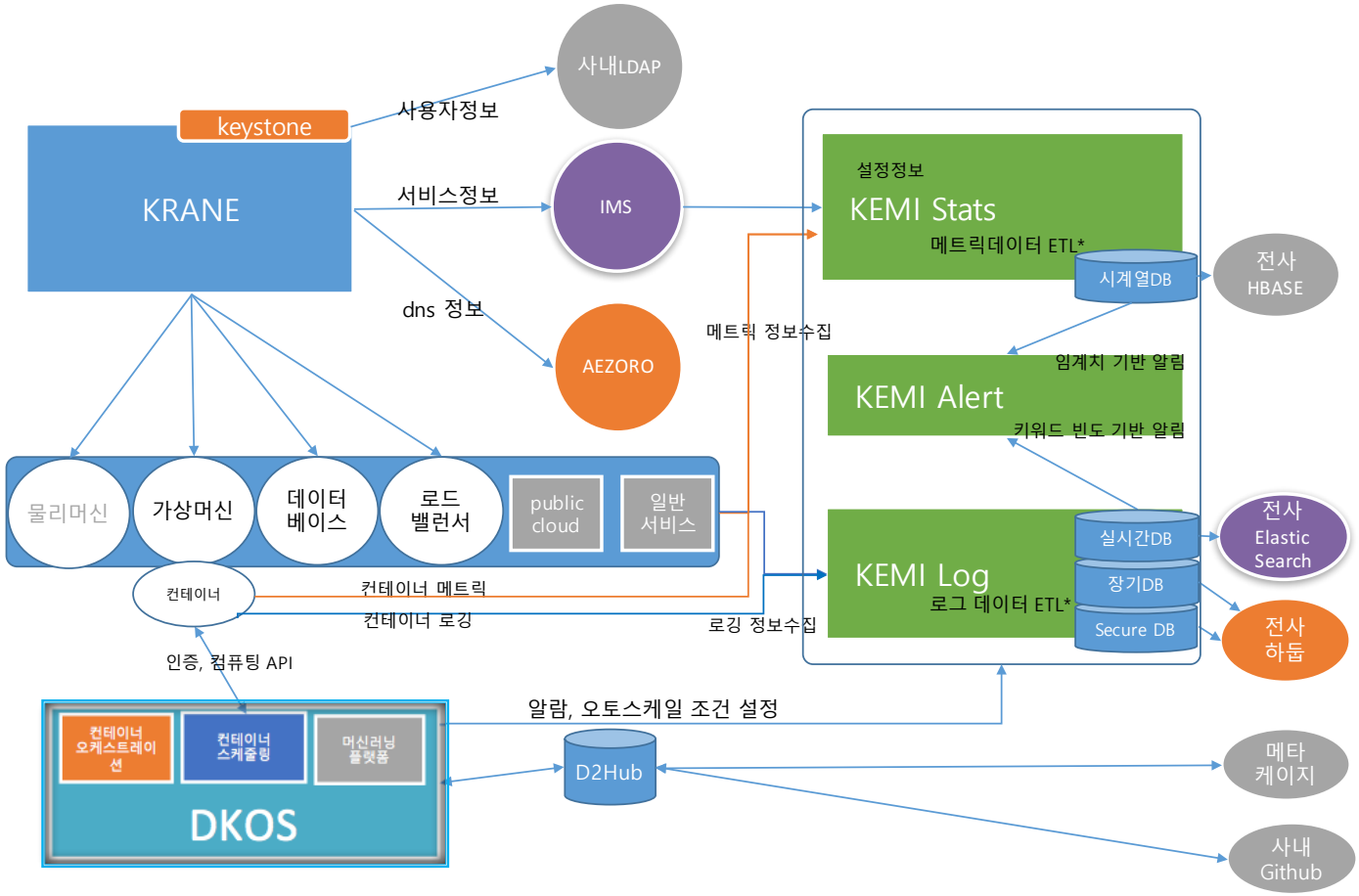
Cloud CMMI3-CaaS

What is the sole purpose of doing CaaS(Container As A Service)?

CMM0	CMM1	CMM2	CMM3
legacy	self service Dev resource	limited Prod resources	Automated CloudUsage
output: cloudTF	output: krane (openstack cloud)	output: kemi (MaaS)	output: DKOS (CaaS)

if (kakao): DKOS(The key.)

DKOS is connected to everything



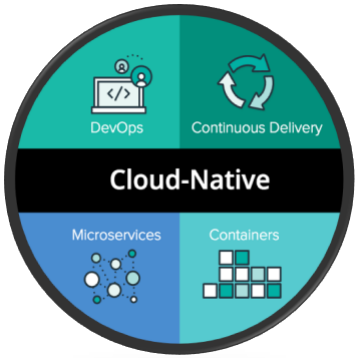
Cloud CMMI4-Cloud Native Platform

What is the sole purpose of doing Cloud Native Platform?

CMM0	CMM1	CMM2	CMM3	CMM4
legacy	self service Dev resource	limited Prod resources	Automated CloudUsage	Integrated Service Platform
output: cloudTF	output: krane (openstack cloud)	output: kemi (MaaS)	output: DKOS (CaaS)	output: 9rum (C.N.P)

Little bit more specific

- Cloud Native Applications Characteristics
  - Micro-services
  - Health Reporting
  - Telemetry Data
  - Resilient
  - Declarative



https://pivotal.io/cloud-native)

↳ First Ask ‘Why we need all these?’

↳ Dynamically Orchestrated Platform

if (kakao):
Characteristics to Requirement

- C. N. Applications Characteristics
  - Microservices
  - Health Reporting
  - Telemetry Data
  - Resiliency
  - Declarative



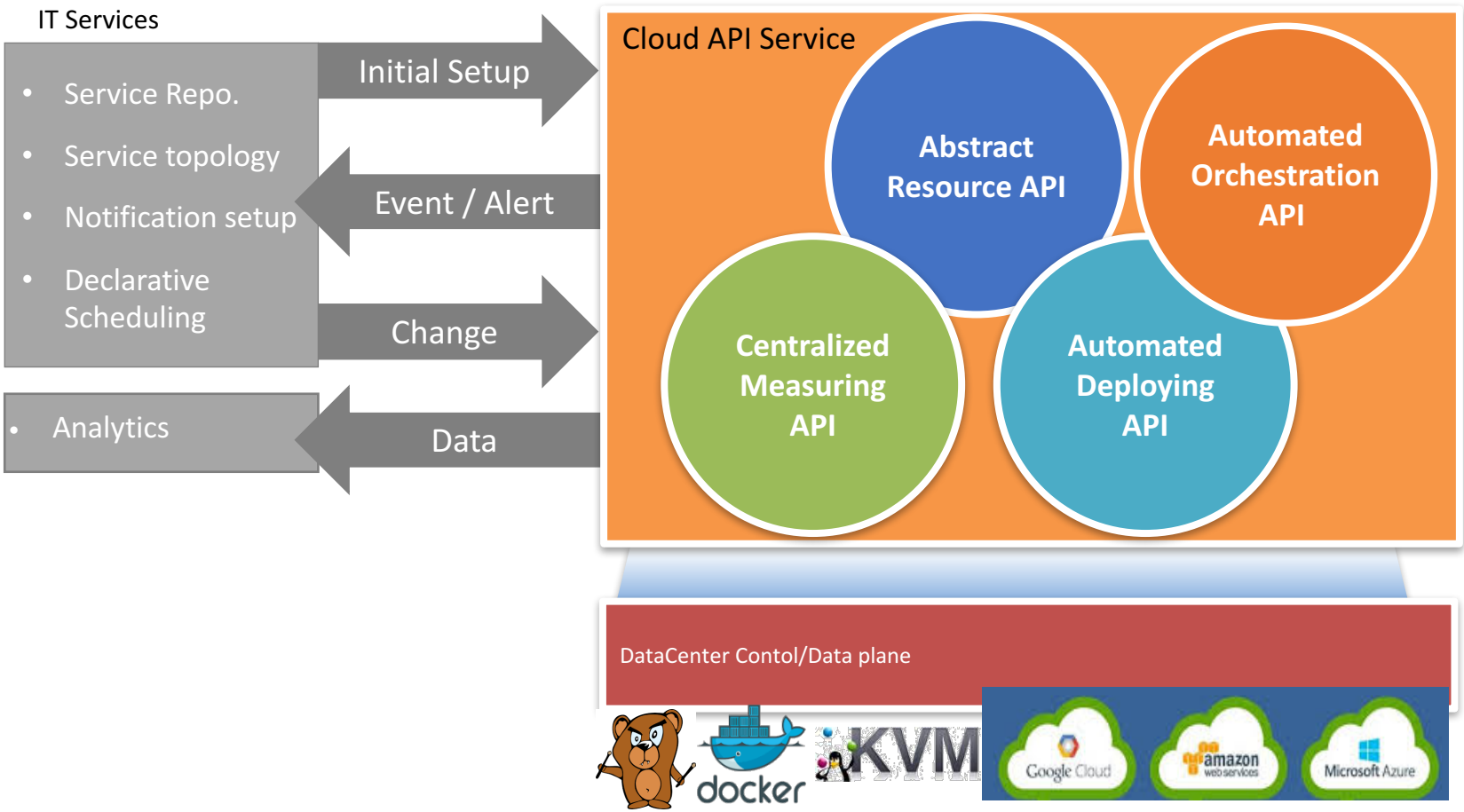
- Platform Requirement
  - Isolation
  - Resource Allocation /Scheduling
  - Service Discovery
  - Monitoring/Logging
  - Metric Aggregation
  - Debugging and Tracing

Cloud CMMI4-Cloud Native Platform

What is the sole purpose of doing Cloud Native Platform?

CMM0	CMM1	CMM2	CMM3	CMM4
legacy	self service Dev resource	limited Prod resources	Automated CloudUsage	Integrated Service Platform
output: cloudTF	output: krane (openstack cloud)	output: kemi (MaaS)	output: DKOS (CaaS)	output: 9rum (C.N.P)

Cloud is heading for Dev can Ops



Some outputs:



Cloud API coverage (%)



Container slave ratio over Cloud (%)



Indexing Data (TB/Day)



Expense $MM/Yr

(compared to public cloud, excluding traffic/storage)



License Expense

Cloud is Just a Tool,
It’s little bit huge though