掃描 QR CODE 填寫問卷
領 AWS Credit 25美元

免費建立 AWS 帳號
即可至攤位領取限量好禮

91APP Way

91APP 在 AWS 上的 SRE 實踐之路

Rick Hwang @ 91APP

2022/04/29

品牌新零售國家隊

2013 年成立 目前員工 500人

1998 年開始投入電商平台經營

電商年資 3,000 人年，OMO 年資 600人年

公司經營規模現況

首家上櫃 SaaS 軟體服務商

代收金流規模近 170 億台幣

累計服務超過 10,000 家零售品牌

服務客戶案例

LVMH 、L'Oréal 、VF品牌集團

全聯、全家、康是美

台灣前五大內衣、過半百大賣家

幫助超過10,000家
中小品牌實現數位轉型

更多案例請上 : 91app.com/showcases/

Rick Hwang

https://www.gtcafe.com
https://rickhw.github.io/

軟體開發者 (Software Developer)、業餘音樂愛好者，工作角色經歷 Software Develpoer、SQA Manager、Ops / Infra Manager、Architect。專長為系統架構、軟體開發、軟體測試、系統維運。這幾年專注在 分散式系統、AWS、DevOps / SRE、經營管理 等領域，著有技術部落格：Complete Think、譯著：分散式系統設計。

工作之餘喜歡科幻小說、金庸武俠、經典文學、哲學、人文藝術。也是業餘音樂愛好者，約有廿年的經驗，涉略涵蓋吉他、鍵盤、編曲、教學，著有音樂部落格：喝咖啡聊音樂 及 電子書 。

Site Reliability Engineering Taiwan

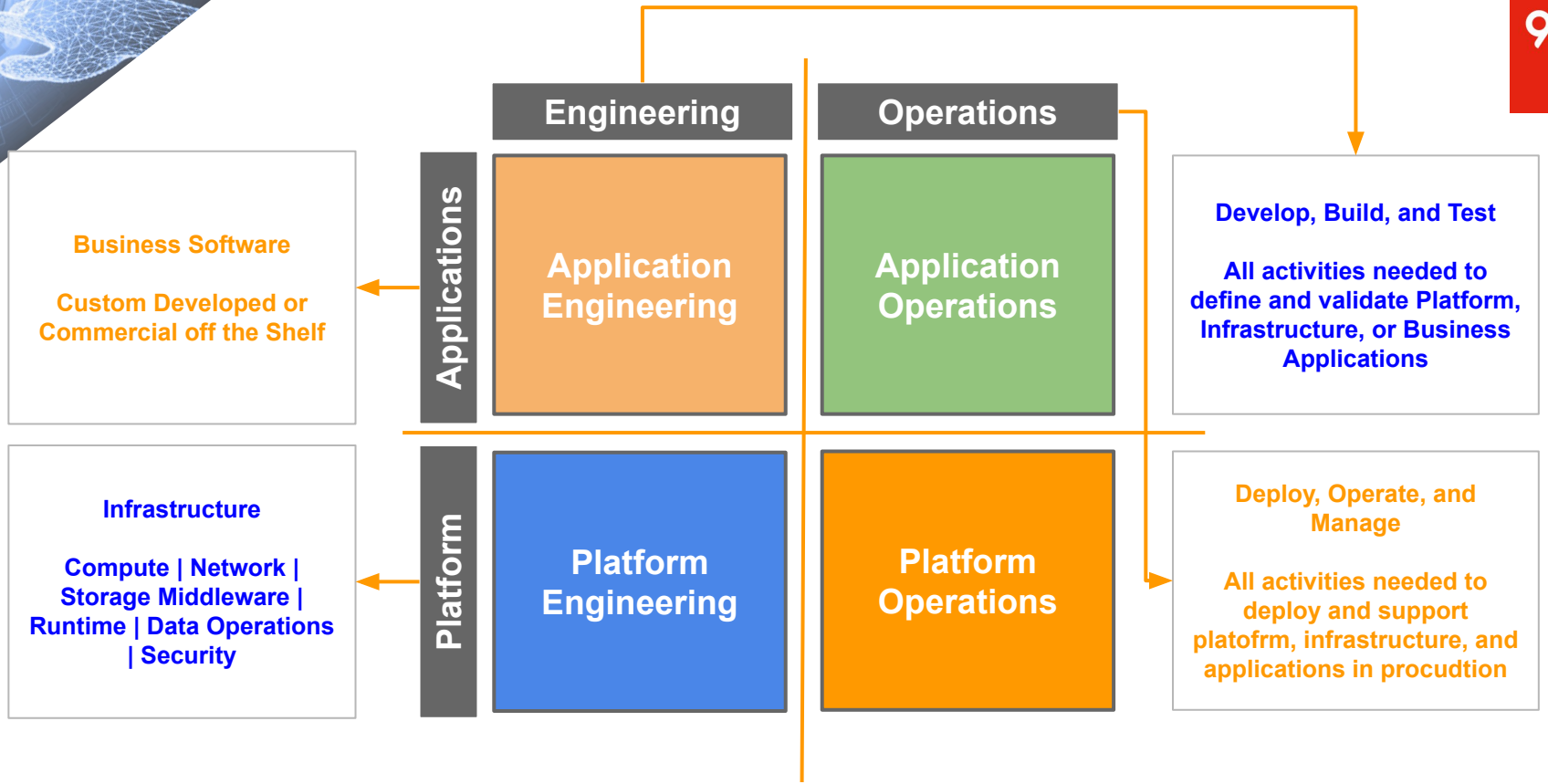
公開社團 · 4,527 位成員

已加入 ＋ 邀請

https://www.facebook.com/groups/sre.taiwan

Agenda

1. How 91APP's SRE implements DevOps using AWS
2. The architecture of SRE as a Service



Source: AWS Well-Architected Framework - Operational Excellence Pillar

91APP

公司的方向

自己開發，自己維運

Feature Teams or AP Team

SRE 要關注的？

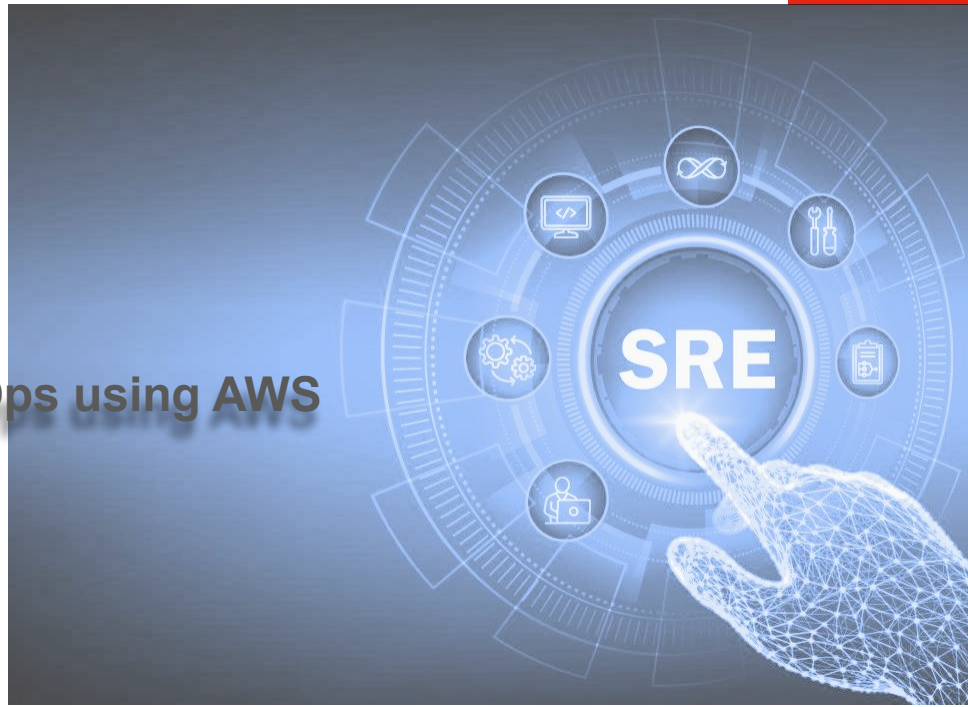
1. 異常處理
2. SLO：監控機制、系統架構
3. 軟體工程：人管系統、系統管服務

91APP

Part 1

How 91APP's SRE implements DevOps using AWS

2015 ~ 2020



問題

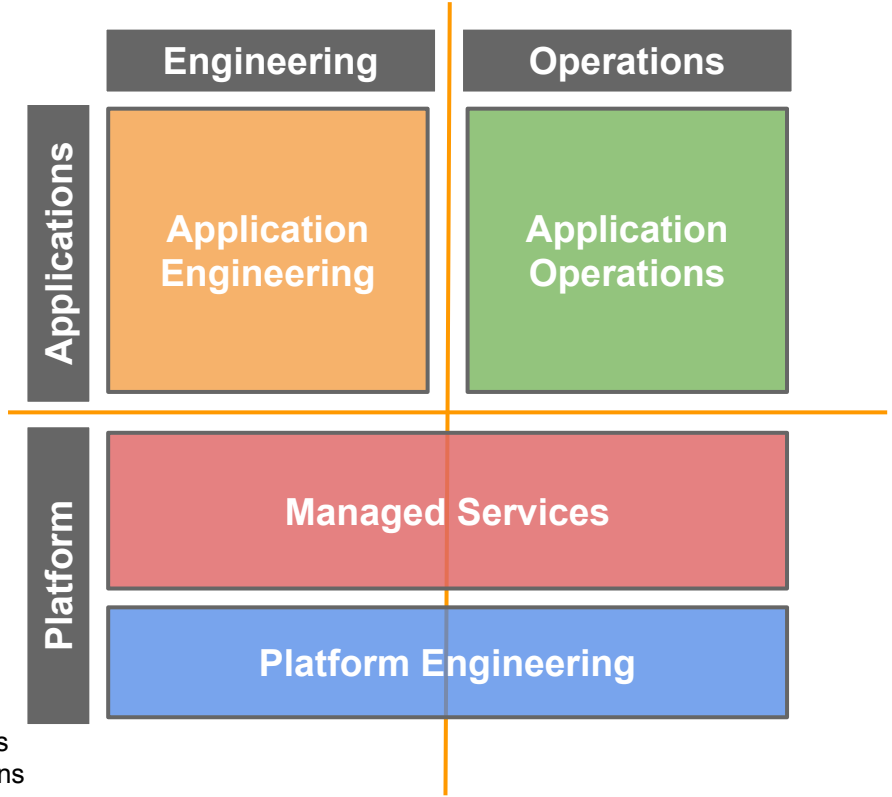
1. 異常處理：從 發生 到 發現，怎麼處理？
    a. 團隊怎麼運作？
    b. 系統架構的資訊一致？
2. 監控機制：工程面（有效的監控）、制度面（值班制度）
3. 系統的可靠度：不定期的業務行銷活動

91APP

我們的 (SRE) 實踐原則：

1. 專注在溝通流程、API 介面
2. 盡可能使用 Managed Services
3. 具體的規範與流程，系統架構透明化
4. 使用軟體工程，讓 SRE 任務規模化

Separated AEO and IEO with Gentralized Governance



AEO: Application Engineering and Operations
IEO: Infrastructure Engineering and Operations

Source: AWS Well-Architected Framework - Operational Excellence Pillar

怎麼做

一、事件處理與管理流程 標準化

二、使用高可靠的監控系統

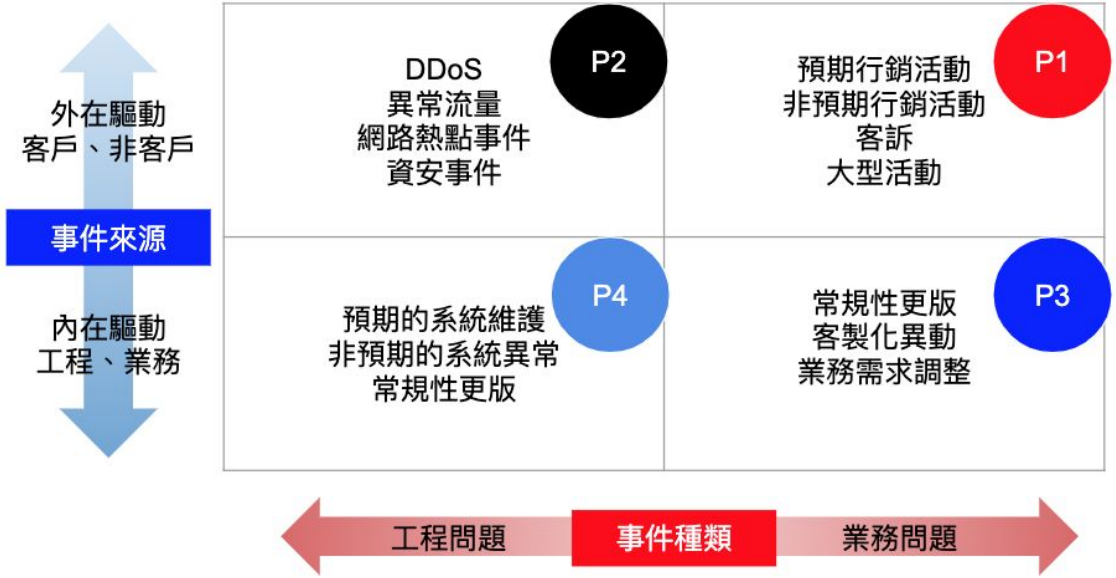
三、提高運行平台的可靠度

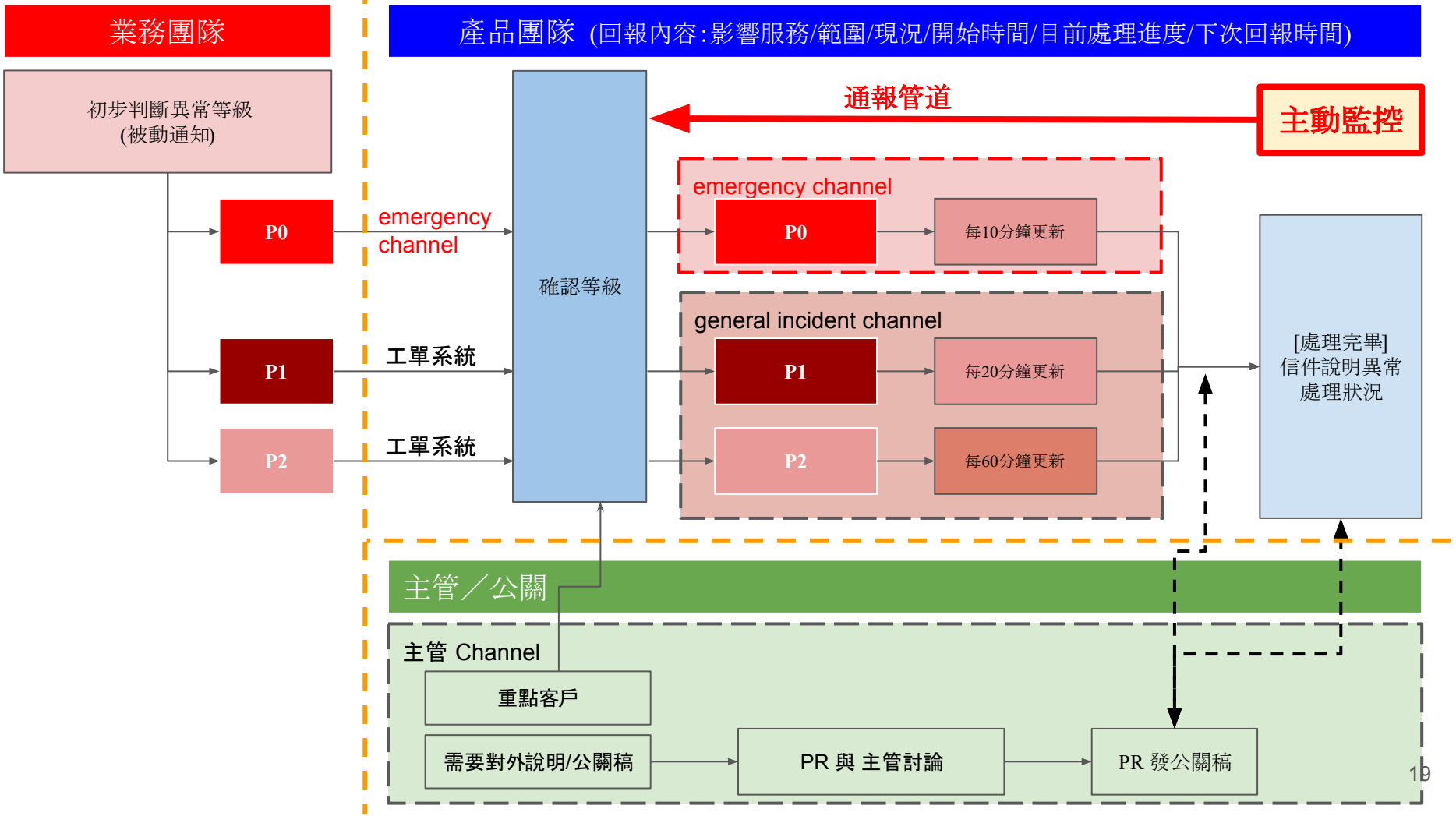
四、降低系統架構的複雜度

五、標準化、規範化

一、事件處理與管理流程 標準化

1. **S1: 客訴的處理**，由**業務維運 (BizOps)** 第一線處理。如果重大活動或者重大客訴，則需搭配**公關與危機管理**程序，擬訂內外應變計畫。
2. **S2: 工程的問題**，由系統維運 (SysOps / SRE) **監控發現**與處理。像是出現異常的流量、被 DDoS 攻擊、突發網路熱點事件 … 等。
3. **S3: 內部工程問題**，這類的事件屬於開發過程中常規性異動，如交付流程與部屬方法沒有處理好，可能會因此造成系統延宕的問題。
4. **P4: 內部維運問題**，屬於維運工程團隊的工作項目，包含自發性的驅動、非預期的驅動。



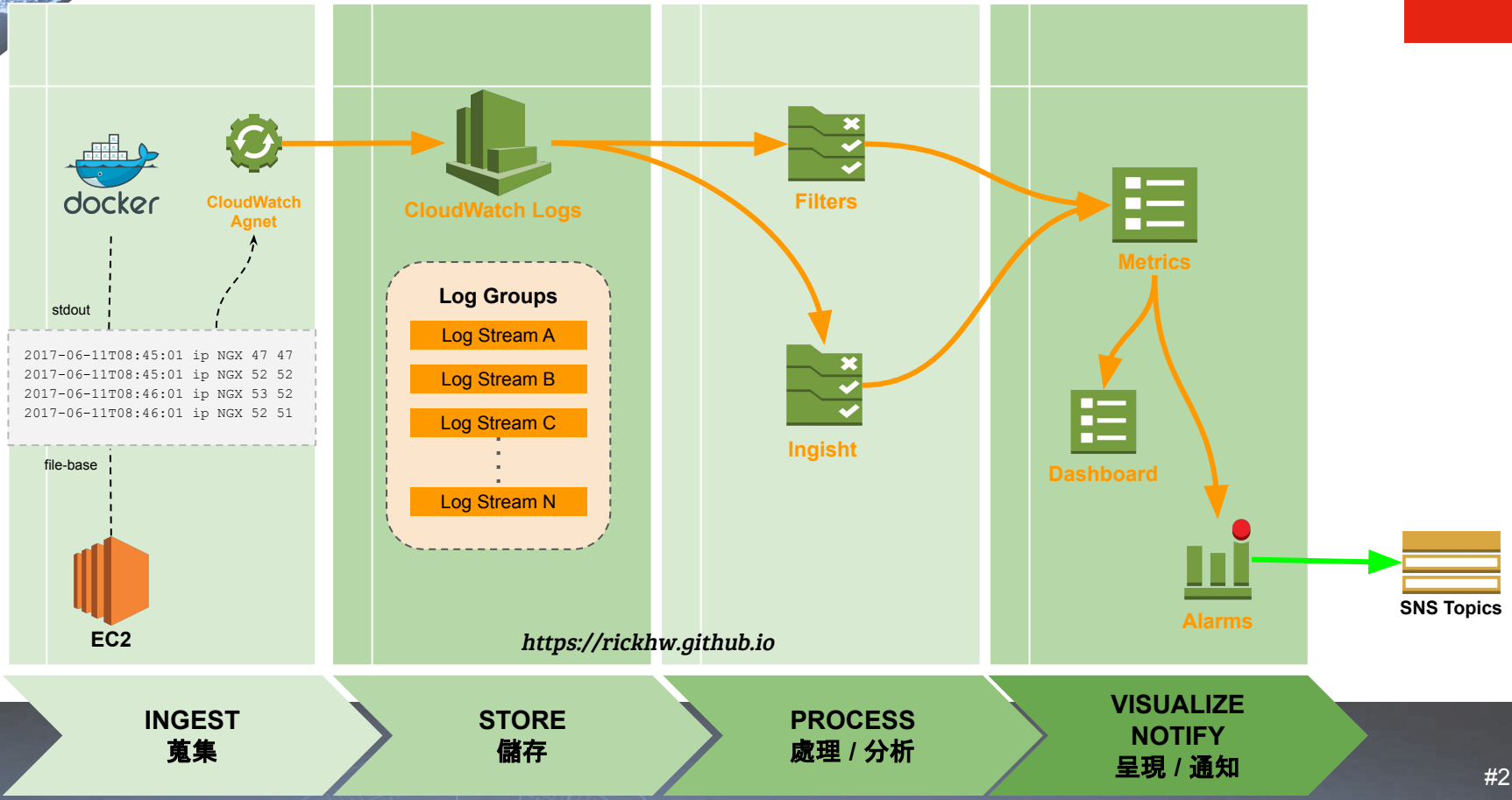

業務團隊
產品團隊 (回報內容:影響服務/範圍/現況/開始時間/目前處理進度/下次回報時間)
初步判斷異常等級
(被動通知)
通報管道
主動監控
P0
emergency
channel
確認等級
emergency channel
P0
每10分鐘更新
P1
工單系統
general incident channel
P1
每20分鐘更新
P2
工單系統
P2
每60分鐘更新
[處理完畢]
信件說明異常
處理狀況
主管／公關
主管 Channel
重點客戶
需要對外說明/公關稿
PR 與 主管討論
PR 發公關稿

二、使用高可靠的監控系統

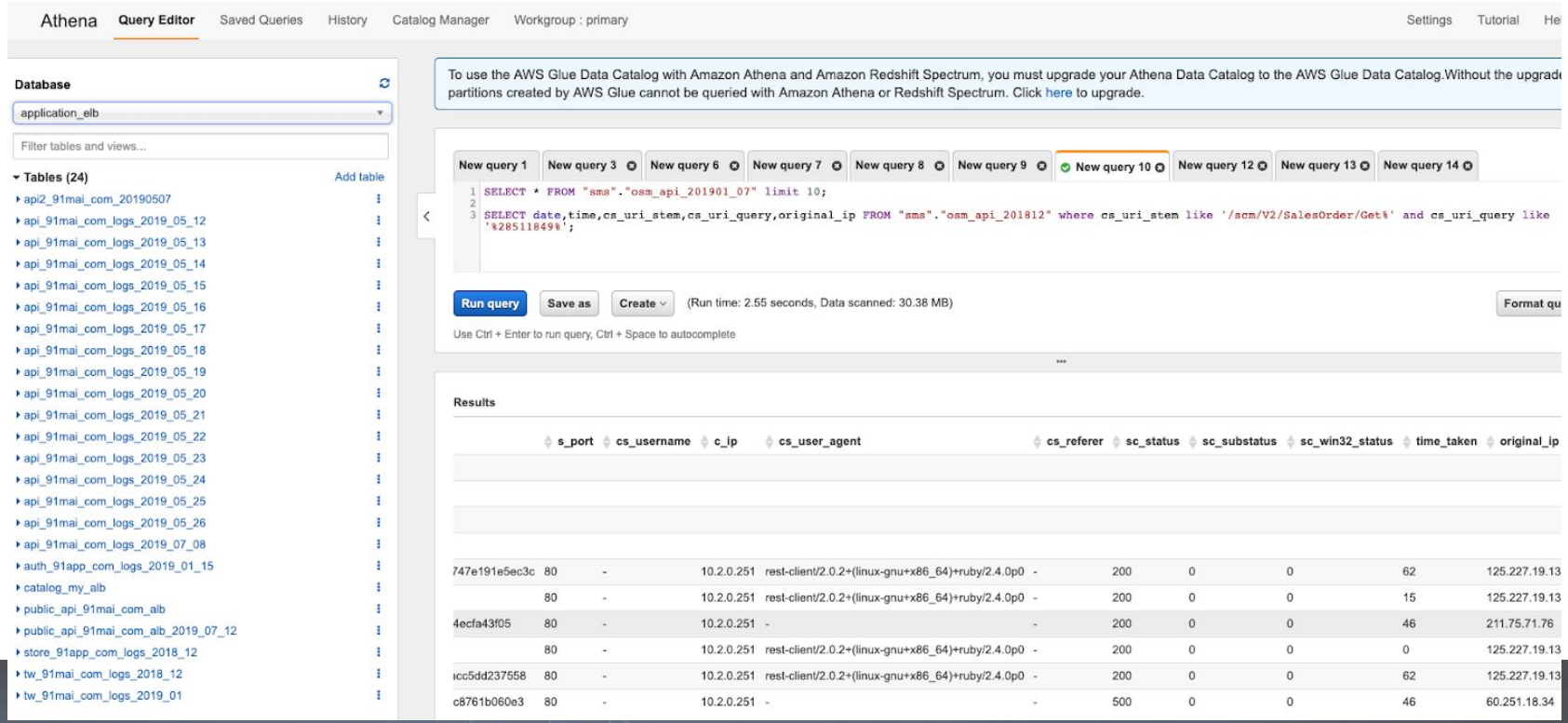
在這個階段，我們的原則

- 技術決策的關鍵：服務業務
- 避免陷入『監控』監控系統的回圈
- 可以接受的成本

AWS CloudWatch 全家桶



Log 分析與查詢：AWS Athena



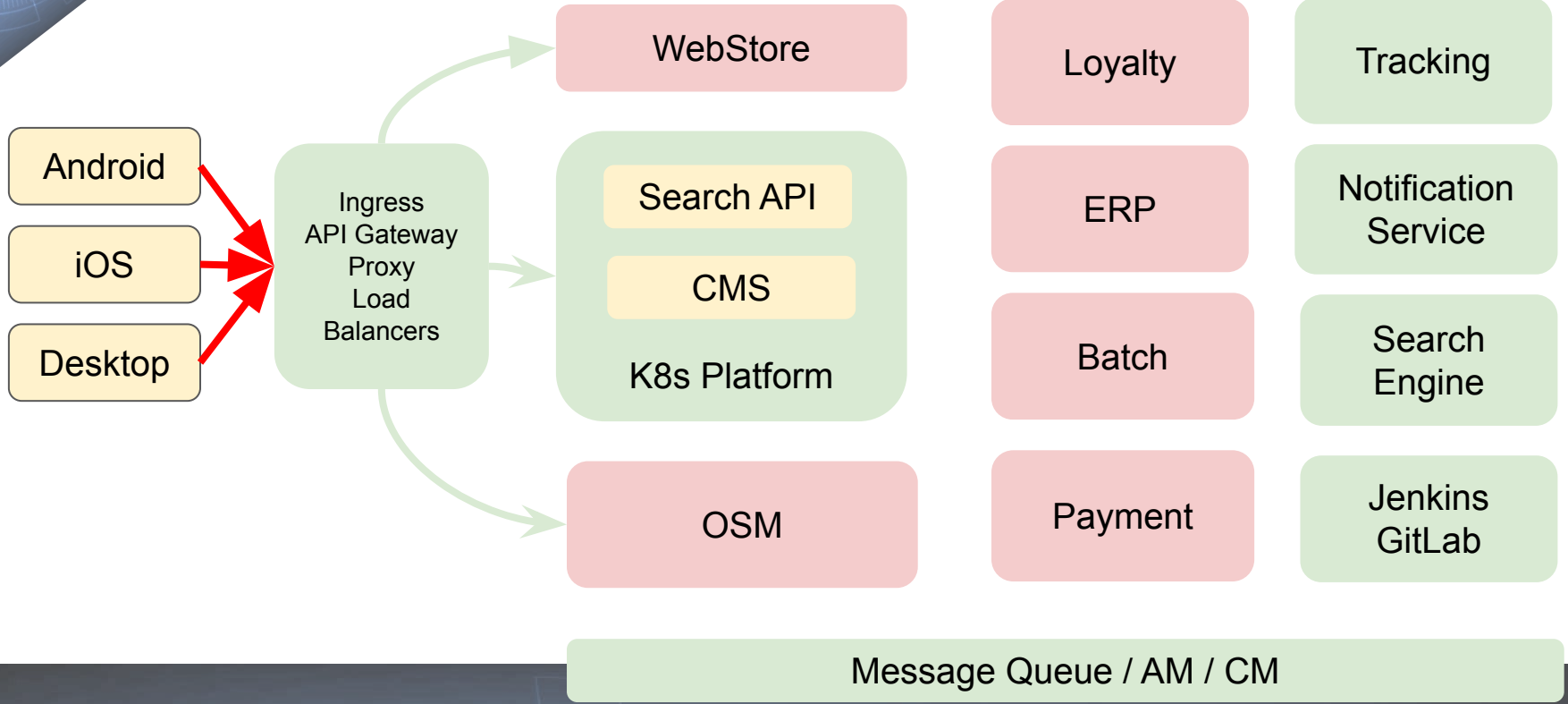
三、提高運行平台的可靠度:導入 K8s，平台化

- 平台 = K8s 本身 (EKS) + 使用者介面 (Rancher) + Dashboard (Grafana)
- 規範資源分配的作業規範，包含成本、申請流程、教育訓練、權限管理、常見問題 … 等。

91APP

四、降低系統架構的複雜度

服務依賴與邊界


Android
iOS
Desktop
Ingress
API Gateway
Proxy
Load
Balancers
WebStore
Search API
CMS
K8s Platform
OSM
Loyalty
ERP
Batch
Payment
Tracking
Notification
Service
Search
Engine
Jenkins
GitLab
Message Queue / AM / CM

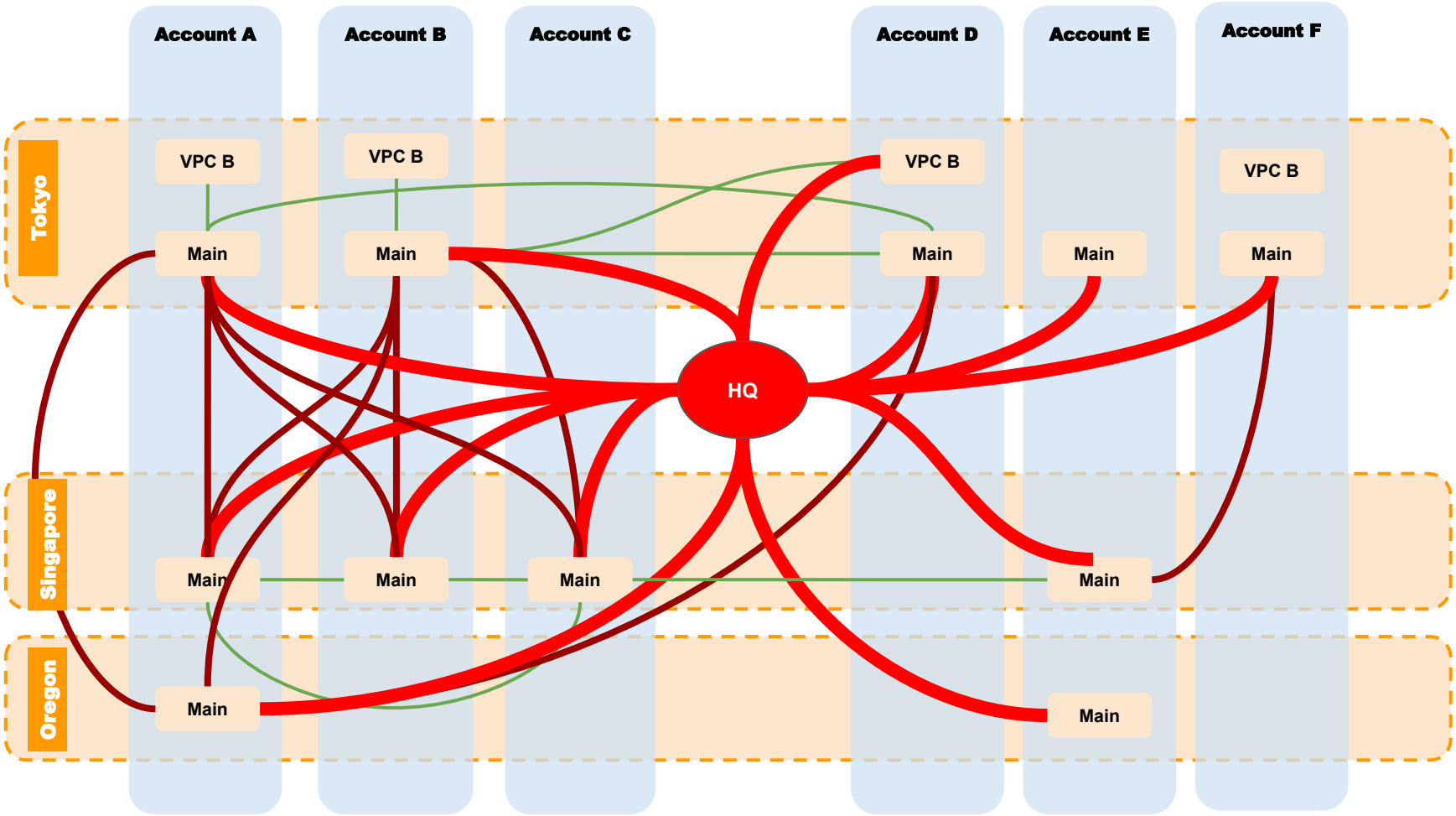
推動基礎架構標準化

- 以 Reliability 為前提，制定**基礎架構標準化**作業
- 標準化是**溝通效率與品質**的依據
- 標準化是**維運自動化**的前置條件
- 架構師與 SRE 共同制定，從上往下落實， 從商業應用，到現場實踐
- 標準化架構的**識別對象** (參考物件導向原則)

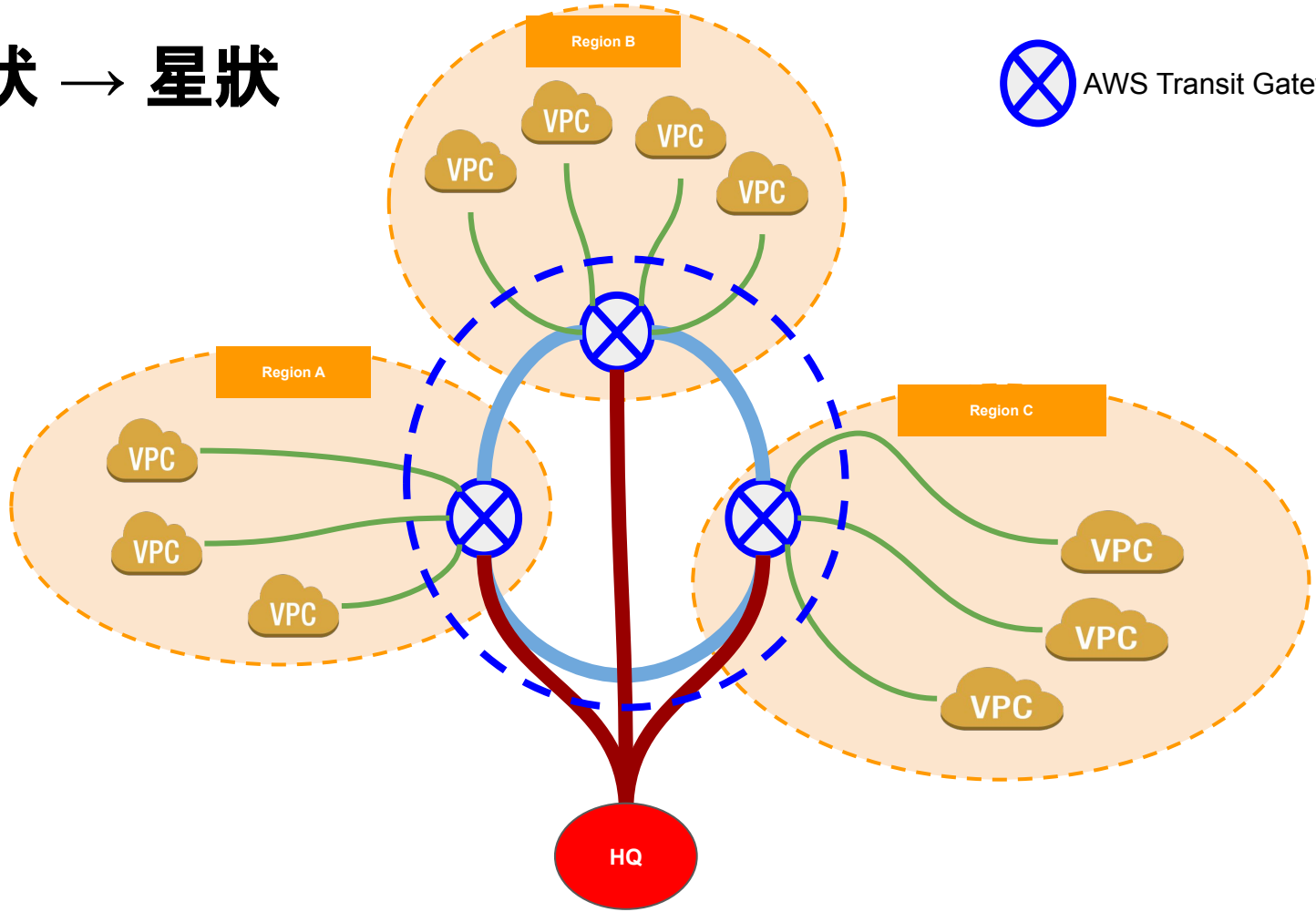
Source: 2018 DevOpsDays 從緊急事件 談 SRE 應變能力的培養

架構規範

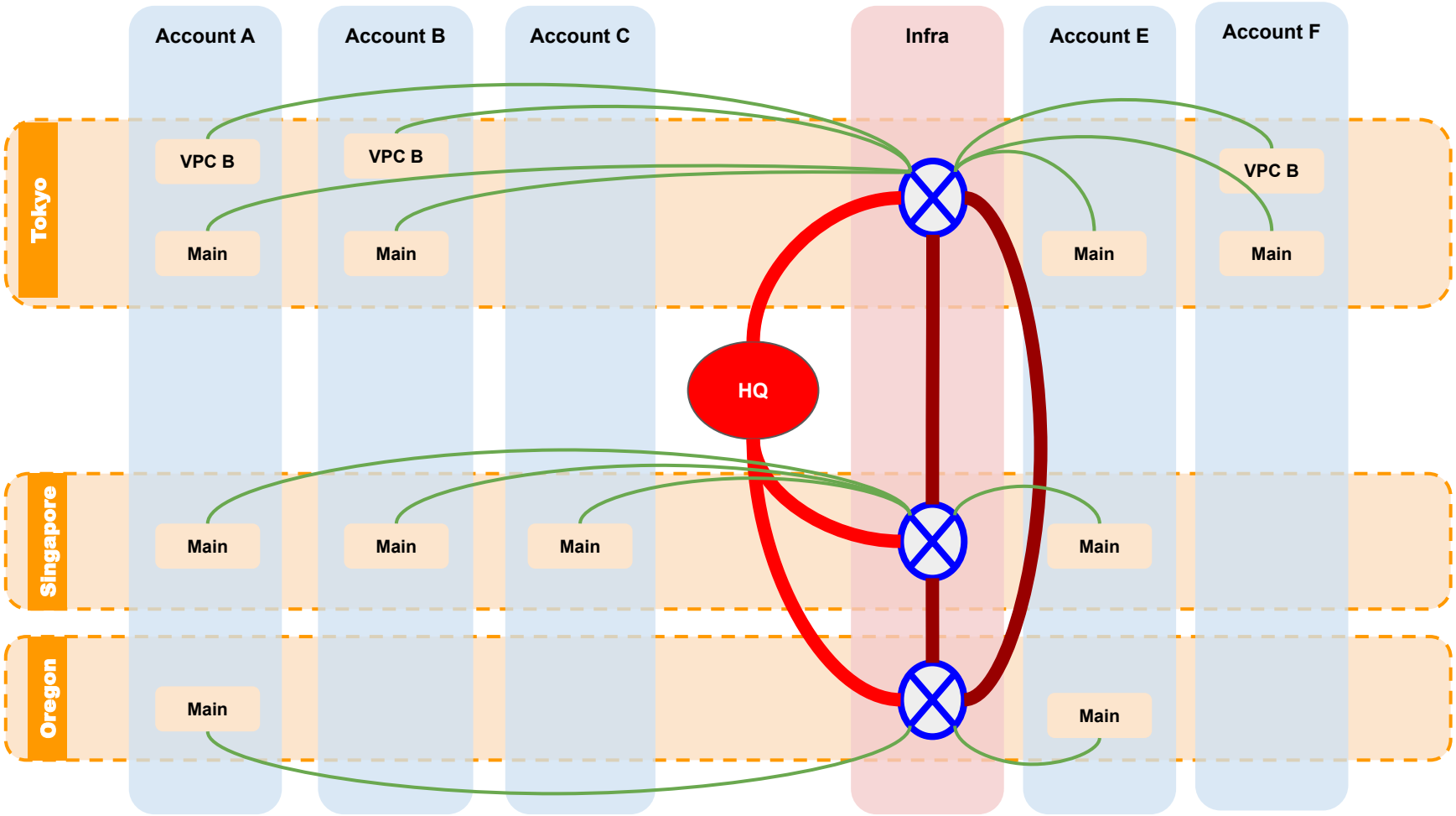
1. **High Level View**: 跳出來，看全景
   - 內外部服務依賴、邊界定義
2. **Service Definition**: 看進去，找本質
   - 抽象角色 (Role) 定義與關係，不談技術
   - 談角色的定義 (Role)、角色與角色之間的依賴關係，Web、DB、API、Batch、LB … 等
   - 角色之間溝通的通訊協議 (HTTP / TCP)、資料流、通訊模式 (主動、被動)
   - 存取控管：Public、Protected、Private
   - 搭配 User Stories
3. **Go Live**: 動手做，找實踐
   - 具象架構實踐，包含測試環境、正式環境的實踐技術 (K8s, AWS or others.)
   - 技術實踐是工程團隊相互溝通用的。
   - 考量可測性，包含環境建置、部署、效能、可測性
   - 考量維運，包含部署、建置、監控、維護、資安、成本、治理。


Account A
Account B
Account C
Account D
Account E
Account F
Tokyo
Singapore
Oregon
VPC B
VPC B
VPC B
VPC B
Main
Main
Main
Main
Main
Main
HQ
Main
Main
Main
Main
Main
Main

目標:網狀 → 星狀



星狀:TX GW -> VPC
網狀:TX GW -> TX GW
星狀:TX GW -> On-Prem


Account A
Account B
Account C
Infra
Account E
Account F
Tokyo
VPC B
VPC B
VPC B
Main
Main
Main
Main
HQ
Singapore
Main
Main
Main
Main
Oregon
Main
Main

五、標準化 / 規範化

開發維運治理

1. 系統架構規範 (Arch Convention)
2. 服務目錄 (Service Catalog)
3. 服務發現 (Service Discovery)
4. 產出物管理 (Artifact Management)
5. 配置管理 (Config Management)
6. 密碼、密鑰管理 (Secret Management)

1. 資源與環境配置 (Provisioning)
2. 建置標準化 (Build Spec)
3. 開發流水線 (Pipeline)
4. 觀測：日誌與監控指標 (Observation)
5. 監控：監控平台與儀表板 (Monitoring)
6. 全域告警系統 (Global Alert System)
7. 異常處理與管理 (Incident Managment)

AWS 資源作業規範

- 認證與授權
  - SSO to AWS Console
  - 跨帳號角色
- Resource Tags
  - 規範原則
  - 全域規範
  - EC2
- IAM 管理規範
  - IAM User
  - IAM Group
  - IAM Role
  - IAM Policy

AWS 資源管理規範

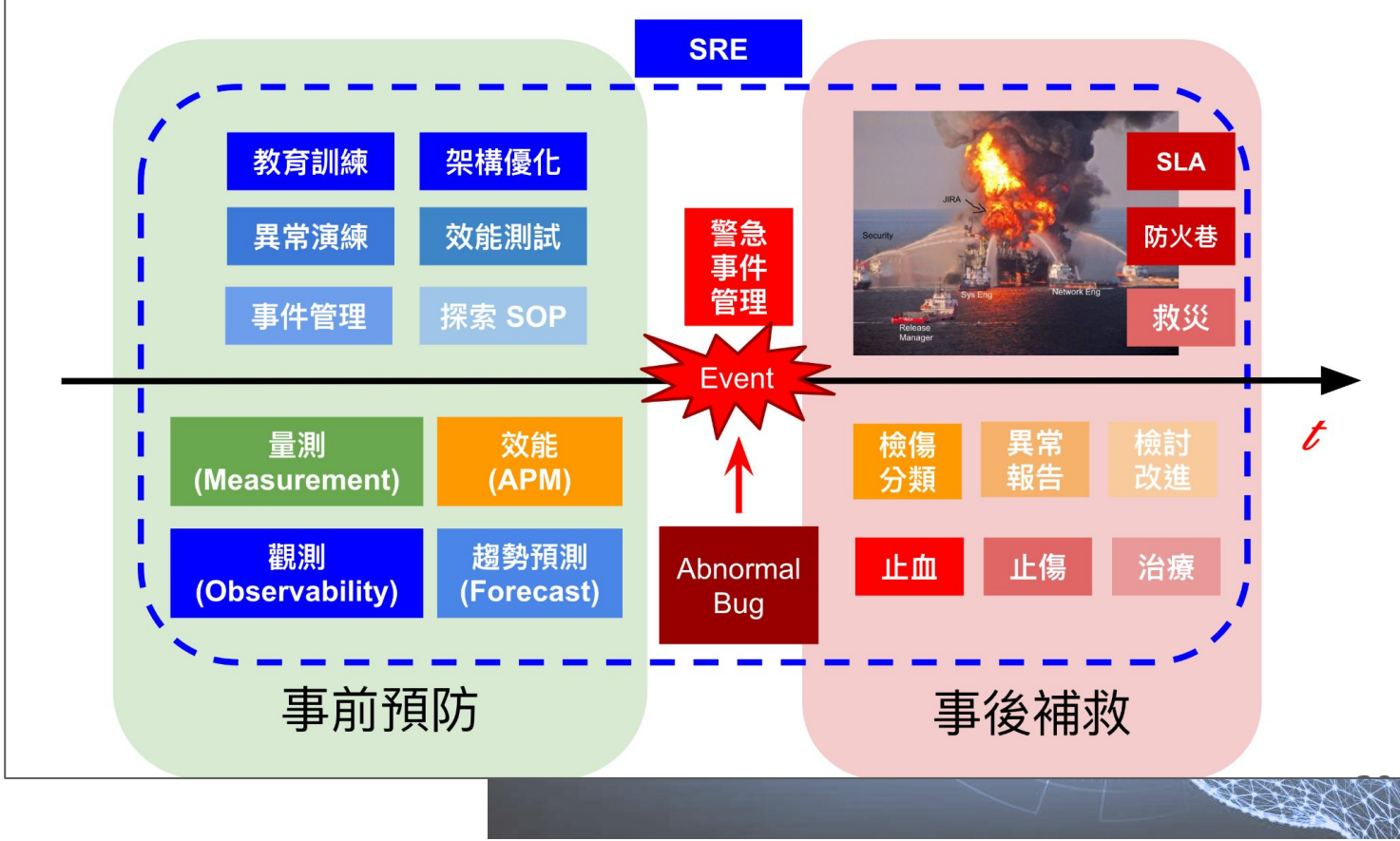
- EC2
- ELB
- VPC
- S3
- CloudWatch
- Security Groups
- DynamoDB
- WAF
- CloudFront
- Athena
- Parameter Store
- Kinesis Data Firehose
- API Gateway

其他:引入其他概念與技術

1. 2018: 邁向 API 經濟 - API Gateway 導入之旅
2. 2019: 聊聊軟體交付的濫觴 談產出物管理
3. 2018: 微服務的基礎建設 - Service Discovery - Andrew Wu
4. 2019: 從零開始的 Configuration Management - Levi Chen
5. 2020: 災難演練 @ AWS 實戰分享 - AWS reInvent reCAP 2019
6. 2020: 在矩陣型組織裡，如何有效管理 AWS 的成本結構與系統架構

91APP

這麼多規範或標準,
有辦法落地？
大家有辦法遵守嗎？



Source: 2018 DevOpsDays 從緊急事件 談 SRE 應變能力的培養

91APP

我們的 (SRE) 實踐原則：

1. 專注在溝通流程、API 介面
2. 盡可能使用 Managed Services
3. 具體的規範與流程，系統架構透明化
4. **使用軟體工程，讓 SRE 任務規模化**

91APP

把戰線，往左移

從開發就開始準備

91APP

Part 2

The archiecture of SRE as a Service

怎麼做

一、事件處理與管理流程

二、使用高可靠的監控系統

三、提高運行平台的可靠度

四、降低系統架構的複雜度

五、標準化、規範化

成長階段

91APP

公司的方向

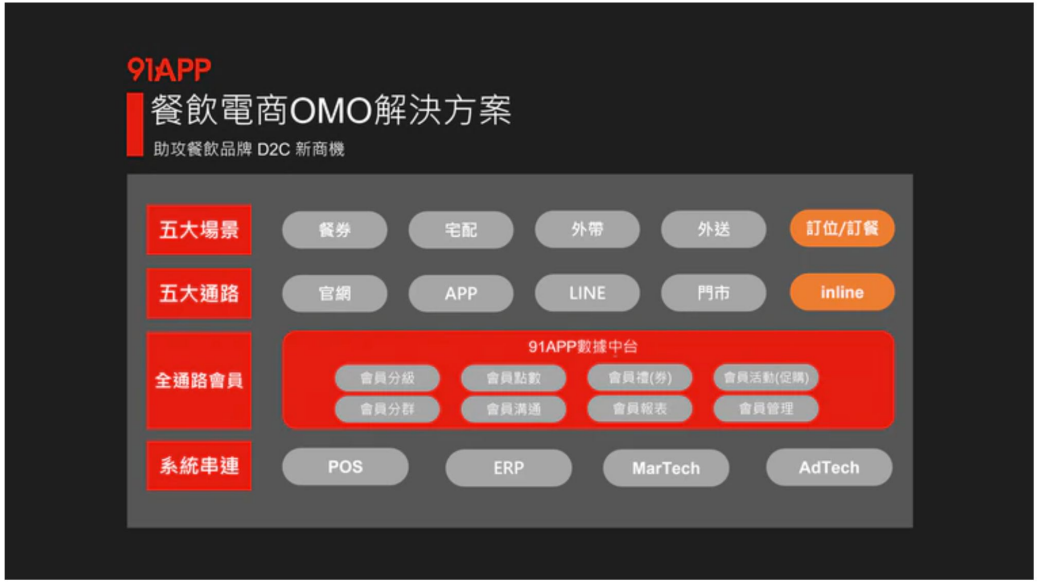
自己開發，自己維運

越來越多 **服務**，越來越多團隊，
面對市場的改變，快速反應。

規範越來越多，怎麼讓規範落地？

與疫共存　91APP推「餐飲電商」助餐飲品牌搶攻百億新商機

17小時前

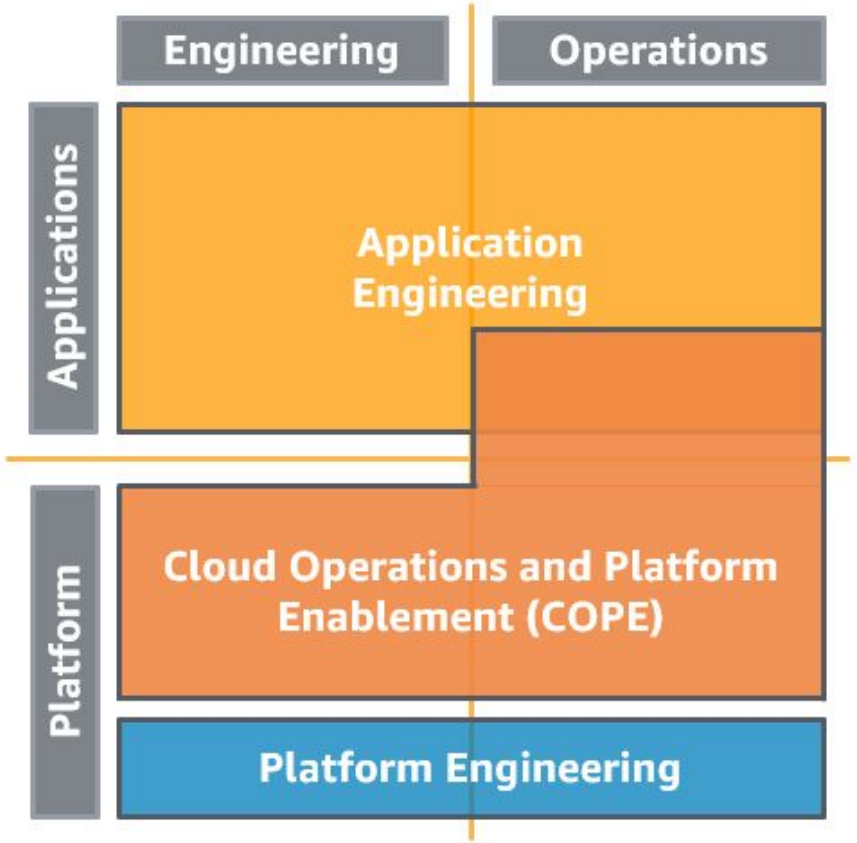


91APP餐飲電商OMO解決方案助攻餐飲品牌D2C新商機。91APP提供

圖片來源：蘋果新聞網

Source: 2022/04/24 - 與疫共存 91APP推「餐飲電商」助餐飲品牌搶攻百億新商機

This “Separated AEO and IEO” model seeks to establish a “**you build it you run it**” methodology.



Source: AWS Well-Architected Framework - Operational Excellence Pillar

SRE as a Service

降低 Application Team 使用 Operations Services 的門檻與複雜度

1. **開發**的制度化、標準化
2. **維運**的制度化、標準化

戰線拉到這裡

這個變成樣板化

怎麼做

一、平台概念藍圖

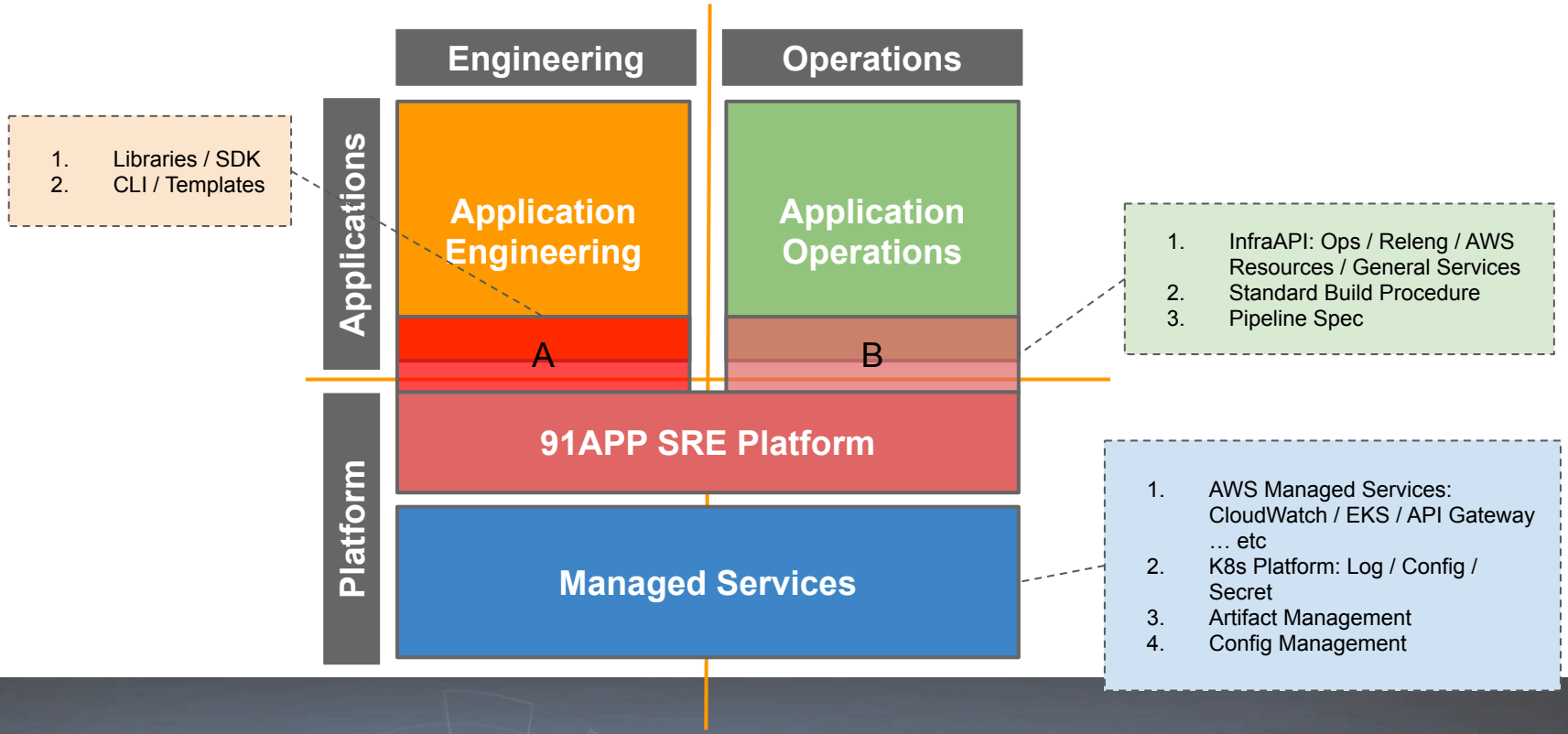
二、Managed Services as API

三、把規範變可實作的規格

四、善用 Managed Services 特性，以 K8s Admission Webhook 為例

五、更多 …

一、平台概念藍圖


Engineering
Operations
Applications
Application Engineering
Application Operations
A
B
Platform
91APP SRE Platform
Managed Services
1. Libraries / SDK
2. CLI / Templates
1. InfraAPI: Ops / Releng / AWS Resources / General Services
2. Standard Build Procedure
3. Pipeline Spec
1. AWS Managed Services: CloudWatch / EKS / API Gateway … etc
2. K8s Platform: Log / Config / Secret
3. Artifact Management
4. Config Management

二、Managed Services as API: InfraAPI

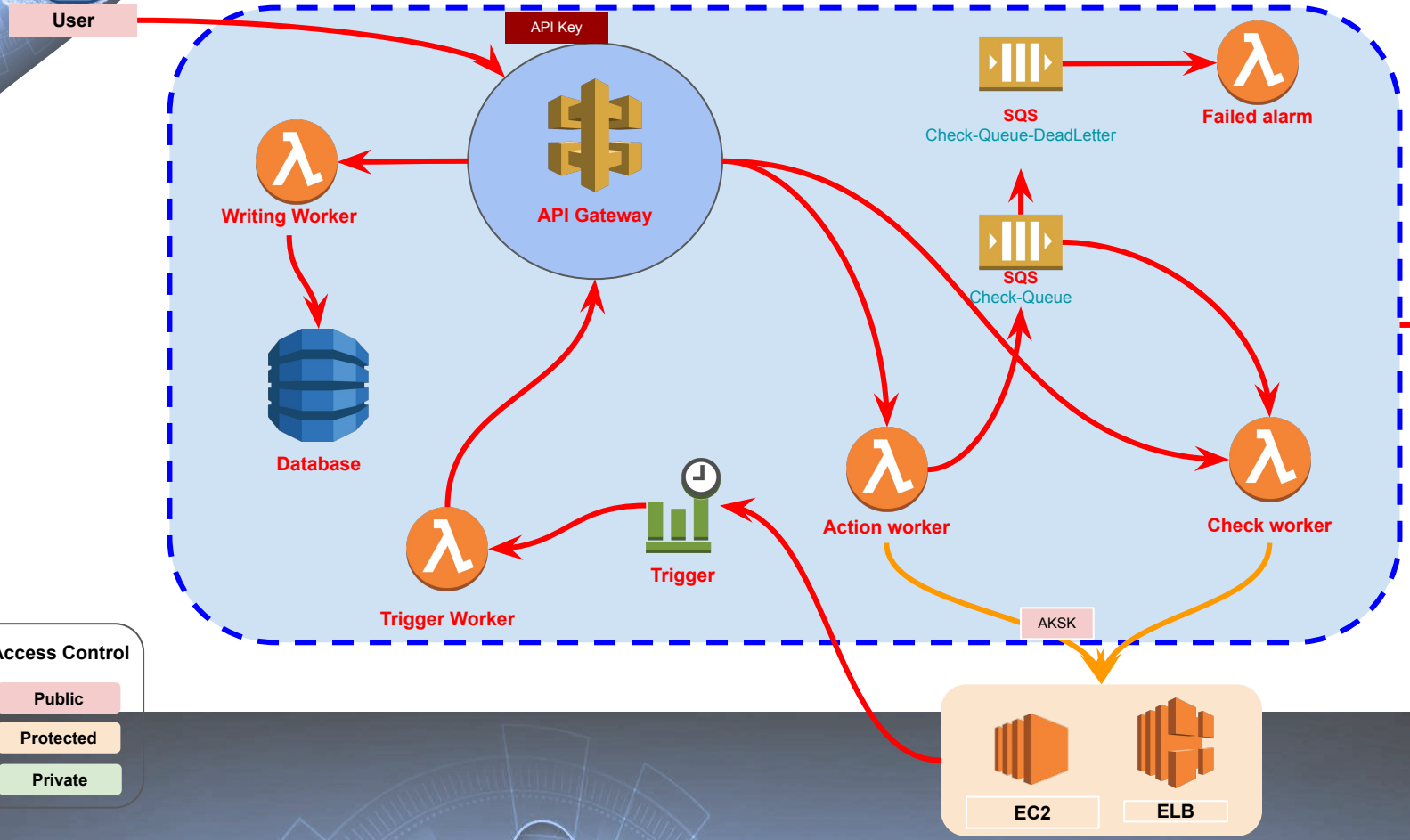
91APP

Ops Operation API, 像是監控, 告警

POST	/Ops/Slack	發送通知訊息給 Slack Channel
POST	/Ops/AWSResource/ELB/Deregister	註銷 instance 從指定的 elbGroup 中
POST	/Ops/AWSResource/EC2/InService	將機器開機且上線服務
POST	/Ops/AWSResource/EC2/OffService	將機器下線並且關機
POST	/Ops/AWSResource/ELB/Register	註冊 instance 到指定的 elbGroup上
POST	/Ops/AWSResource/EC2/Start	將指定 instance 開機
POST	/Ops/AWSResource/EC2/Stop	將指定 instance 關機

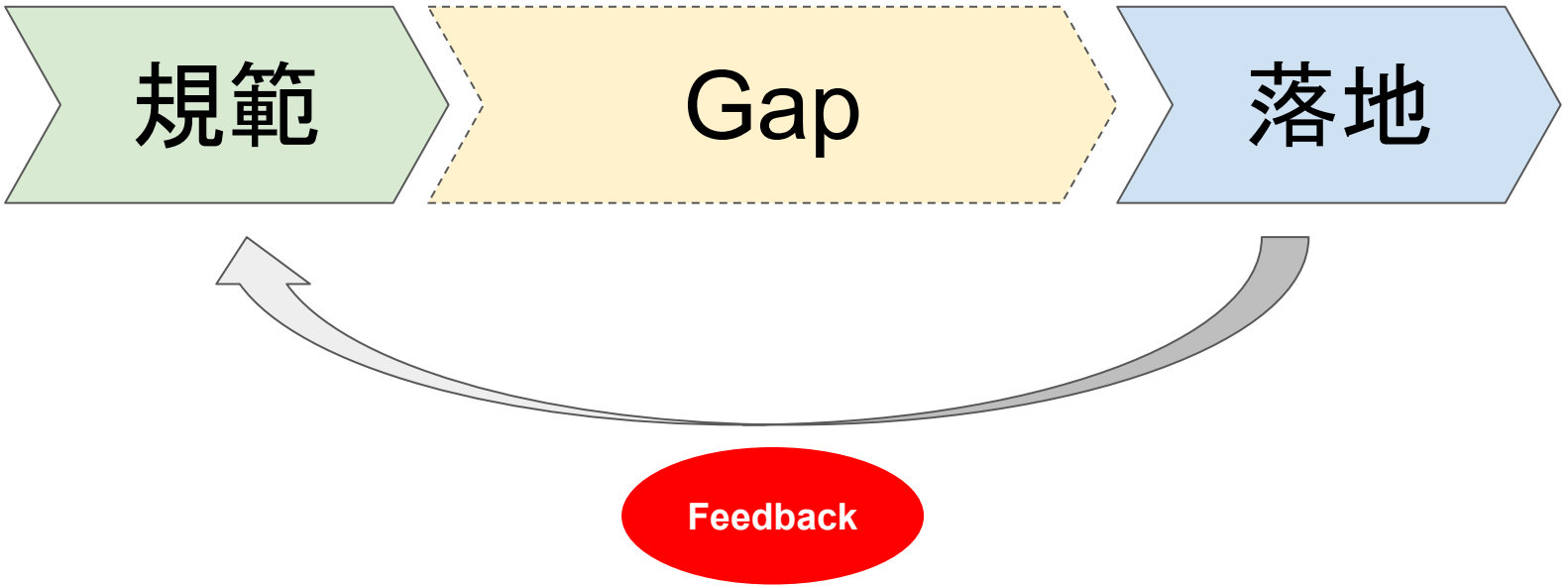
POST	/Ops/AWSResource/ELB/AddRuleQuarantine	新增 Listener Rule 規則
POST	/Ops/AWSResource/EC2/TerminatedUpdateTable	將被 terminate 機器記錄起來
POST	/Ops/AWSResource/ELB/GroupInstancesState	查看 ELB Group 目前有什麼機器
POST	/Ops/AWSResource/ASG/CreateLaunchTemplate	【開發中】產生 AutoScaling
GET	/Ops/AWSResource/ASG/LaunchTemplateList	【開發中】列出所有 LaunchTem
POST	/Ops/AWSResource/ASG/SetDesiredCapacity	【開發中】將指定的 ASG 長出 d
POST	/Ops/AWSResource/ASG/SetWeight	【開發中】調整 ELB Group 流量比重
GET	/Ops/AWSResource/ASG/GroupWeight	【開發中】查看 ELB group weight

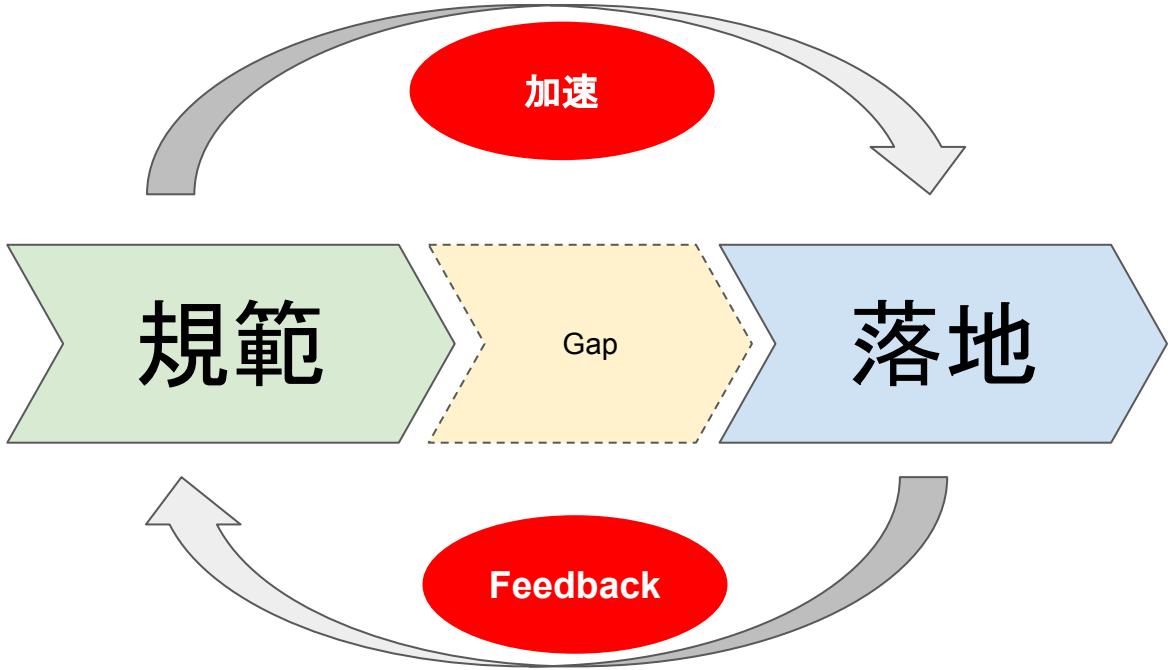
InfraAPI: Global Alert System



91APP

三、把規範變可實作的規格


規範
Gap
落地
Feedback

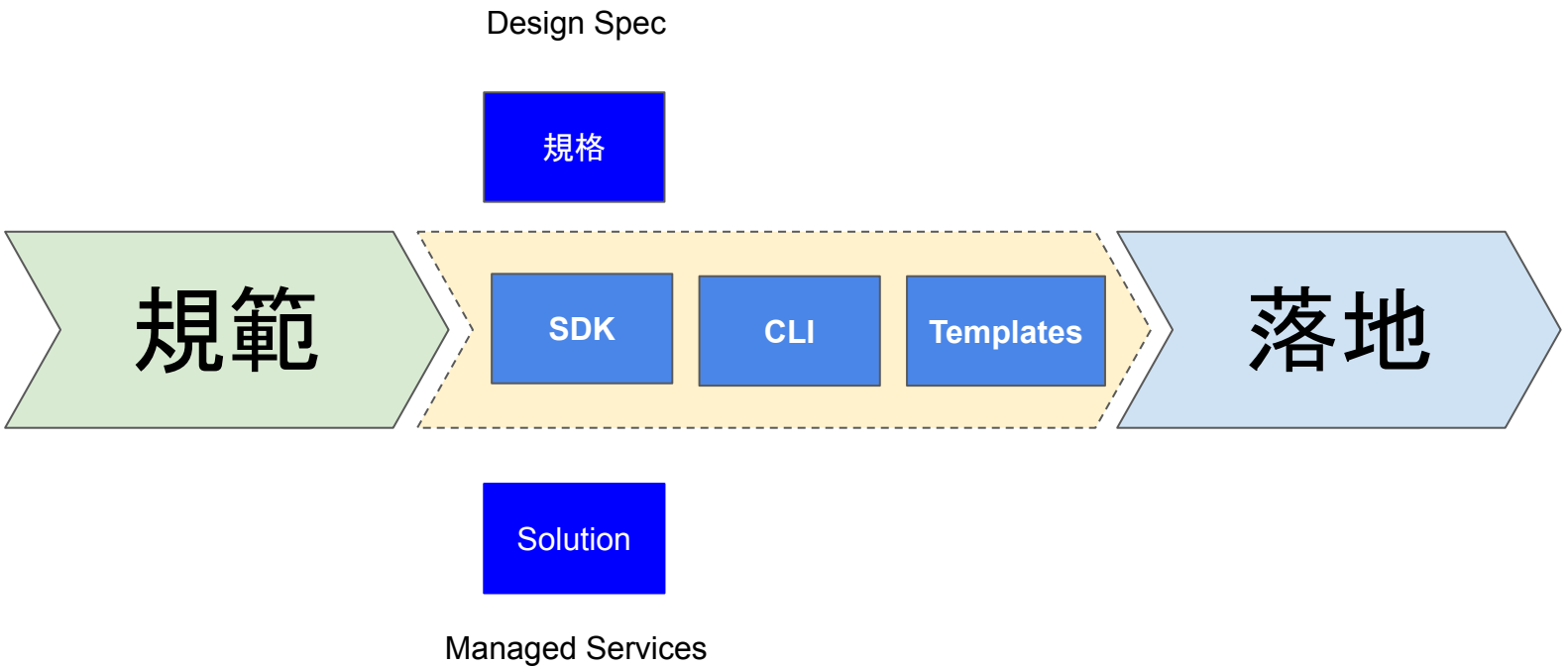

加速
規範
Gap
落地
Feedback

讓**規範**落地的**三個角色**

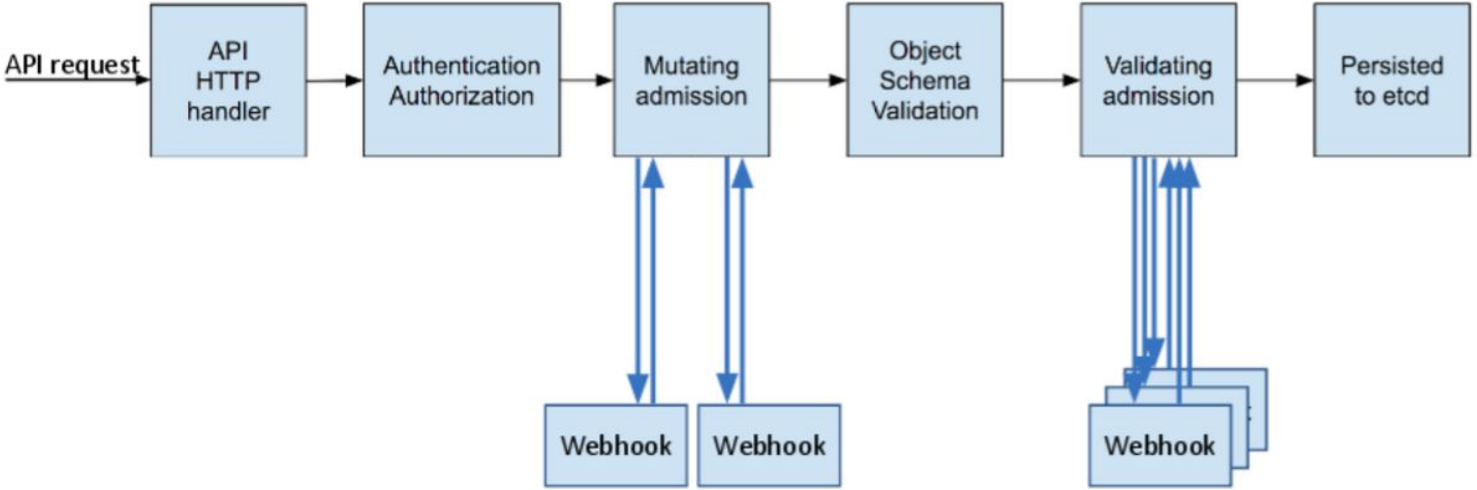
1. 專注在分析需求，讓規範變成規格的人
   a. RFC，可以用任意程式語言實踐，用同樣的方法驗收
2. 用技術實踐規格的人
   a. 找 System Solution 或者自己幹 - Infra
   b. 封裝規格：Libraries -> Framework -> Platform
3. 管理決策與資源協調者
   a. 管理進度、溝通協調
   b. 收需求問題、資源協調

實作的原則

1. **Spec First**：把需求轉化成可以跨語言、跨平台的實作規格，類似於 RFC
2. **API 化**：封裝 Managed Services 的複雜度，降低使用門檻
   - 封裝 AWS 常用服務，像是 EC2 開關機、上下 ELB、Service 狀態
3. **SDK / Lib / Framework**：給 Application 介接 Solution 的標準介面
   - Config、Log、Secret、Network、Security、Standard Data Model
4. **Tools**：
   - CLI：包含 SSO、Code Templates、Build Spec … etc.
   - Templates: Code Templates (.NET / PHP / node.js … etc)、Pipeline Template、Observation Templates、Architect Templates (IaC)
   - IDEilize：VSCode Extension

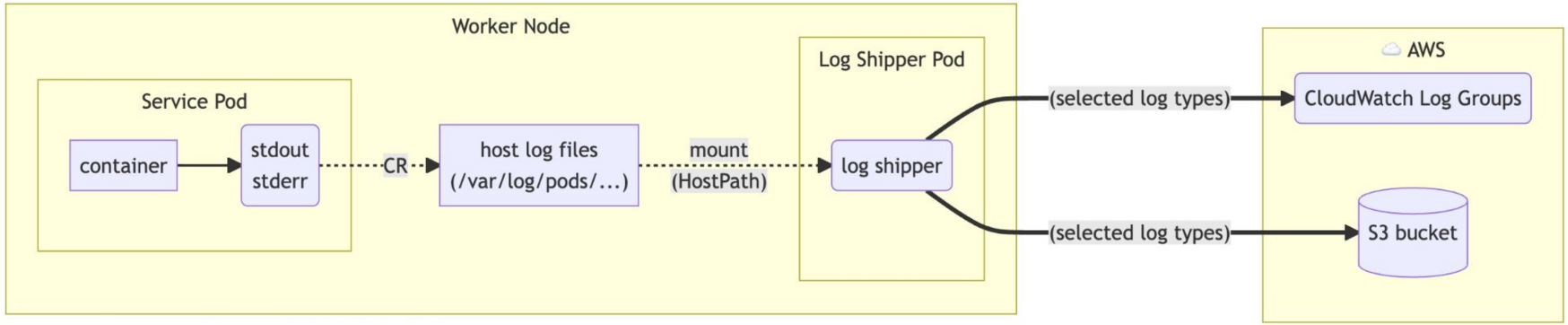

Design Spec
規格
規範
SDK
CLI
Templates
落地
Solution
Managed Services

四、善用 Managed Services 特性：K8s Admission Webhook



- Admission Controller 會攔截經過 authentication & authorization 之後的 API request

Log Shipping by Sidecar Pattern



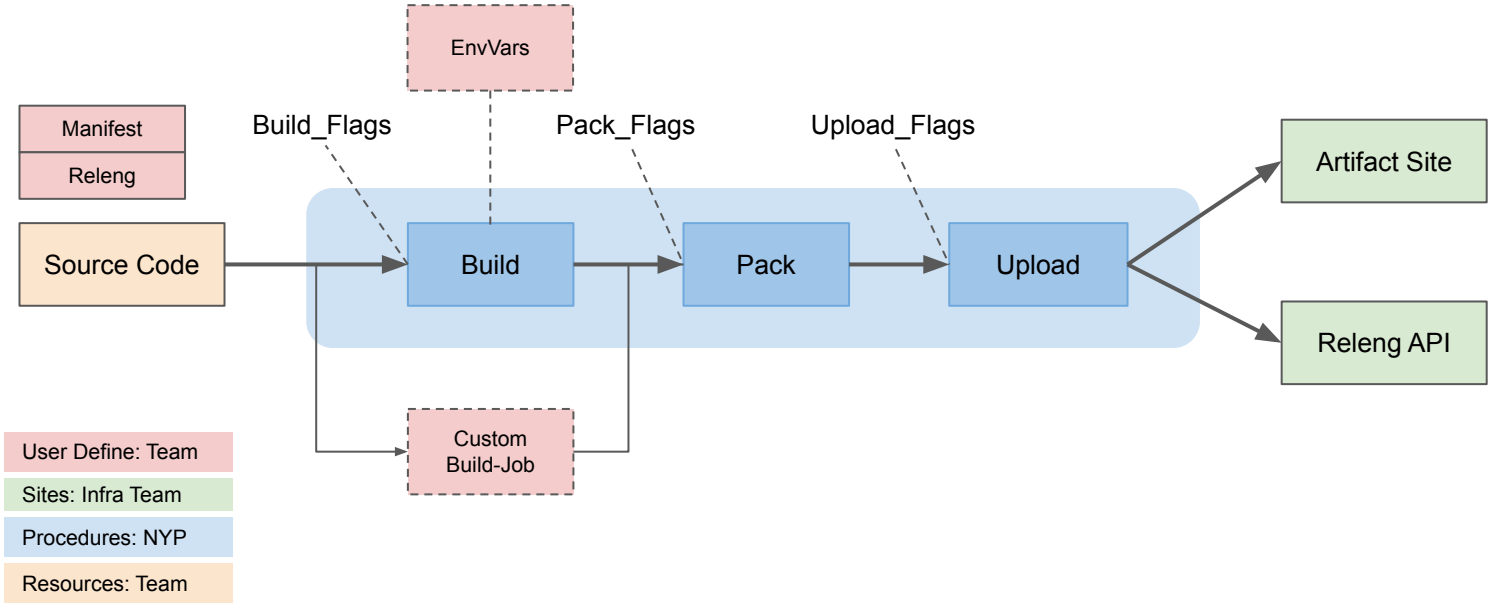
```yaml
template:
  metadata:
    labels:
      app: random-service
    annotations:
      # 固定用法，填 true 即可
      logging.91app.io/inject: "true"
      # 請參考欄位說明 #1
      logging.91app.io/exec.<CONTAINER_NAME>: "sh -c tail -f /dev/null"
      # log key，沒意外的話會是 "_lt"
      logging.91app.io/logKey: "_lt"
      # 要送到 CloudWatch Logs 的 log type。Infra team 會提供，請參考欄位說明 #2
      logging.91app.io/output.cloudwatch: "metrics"
      # Infra team 會提供，每個環境的值都不一樣，請參考欄位說明 #2
      logging.91app.io/output.s3: "[redacted]domService"
      # 請參考欄位說明 #3
      logging.91app.io/registry: "[redacted]"
```

五、開發維運治理

1. 系統架構規範 (Arch Convention)
2. 服務目錄 (Service Catalog)
3. 服務發現 (Service Discovery)
4. 產出物管理 (Artifact Management)
5. 配置管理 (Config Management)
6. 密碼、密鑰管理 (Secret Management)

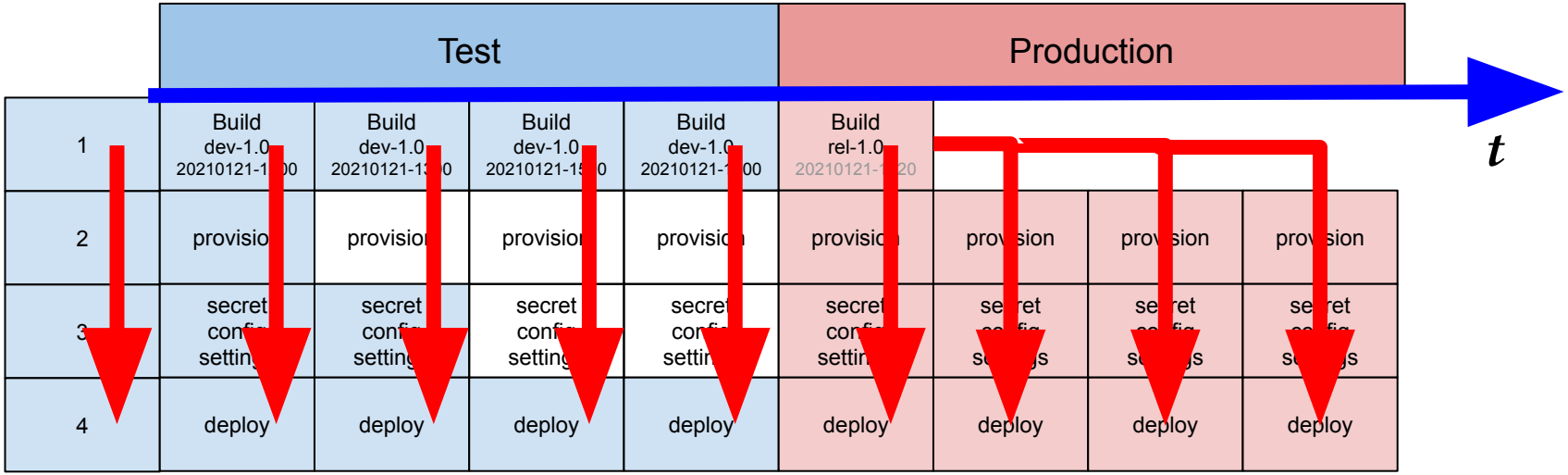
1. 資源與環境配置 (Provisioning)
2. 建置標準化 (Build Spec)
3. 部署標準化 (Deploy Spec)
4. 開發流水線 (Pipeline)
5. 觀測：日誌與監控指標 (Observation)
6. 監控：監控平台與儀表板 (Monitoring)
7. 全域告警系統 (Global Alert System)
8. 異常處理與管理 (Incident Managment)

Standard Build Procedures / Spec



universal build spec for: docker, nuget, npm, raw, … etc.

Standard Deployment Spec




```
~$ n1cli
  _   _   _                    _     ____   _   _
 | \ | | (_)  _ __     ___    / |   / ___| | | (_)
 |  \| | | | '_ \   / _ \ | | | |     | | | | |
 | |\  | | | | | | |  __/ | | | |___  | | | | |
 |_| \_| |_| |_| |_|  \___| |_|  \____| |_| |_|
USAGE:
    n1cli [OPTIONS] <COMMAND>
OPTIONS:
    -h, --help       Prints help information
    -v, --version    Prints version information
COMMANDS:
    login       登入內部系統
    new         建立新專案
    build       建制專案
    deploy      部署服務
    update      檢查是否有可以更新的版本
    upgrade     進行版本更新
    version     檢視版本
```

91APP

~$ n1cli deploy {environment-x} {blue-green-flow}

Environments:
1. Secret
2. Config
3. Env Vars

Flows:
1. Pre-procedure
2. Template rendering
3. Deploy process
4. Post-procedure

很多 ...

1. 系統架構規範 (Arch Convention)
2. 服務目錄 (Service Catalog)
3. 服務發現 (Service Discovery)
4. 產出物管理 (Artifact Management)
5. 配置管理 (Config Management)
6. 密碼、密鑰管理 (Secret Management)

1. 資源與環境配置 (Provisioning)
2. **建置標準化 (Build Spec)**
3. **部署規格 (Deploy Spec)**
4. 開發流水線 (Pipeline)
5. 觀測：日誌與監控指標 (Observation)
6. 監控：監控平台與儀表板 (Monitoring)
7. 全域告警系統 (Global Alert System)
8. 異常處理與管理 (Incident Managment)

規範 -> 規格 -> Solution / Tools

SRE as a Service

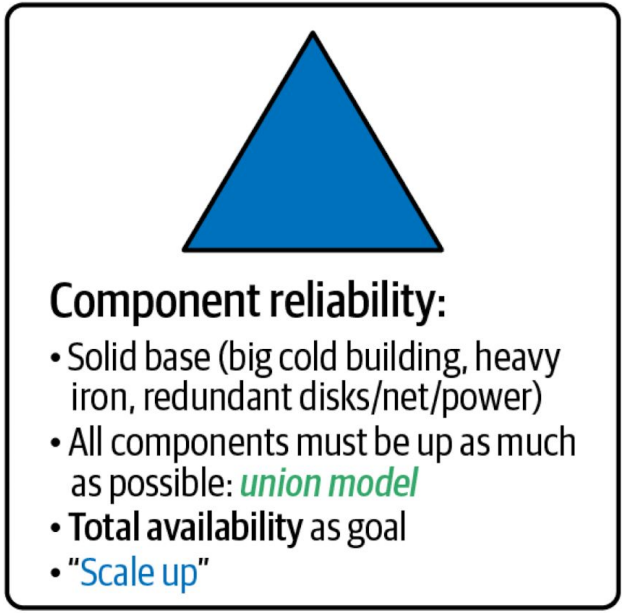
降低 Application Team 使用 Operations Services 的門檻與複雜度

1. 開發的標準化：Config、Log、Secret、Network、Security … etc
2. 維運的標準化：Build、Pipeline、Deployment、Observation … etc

91APP

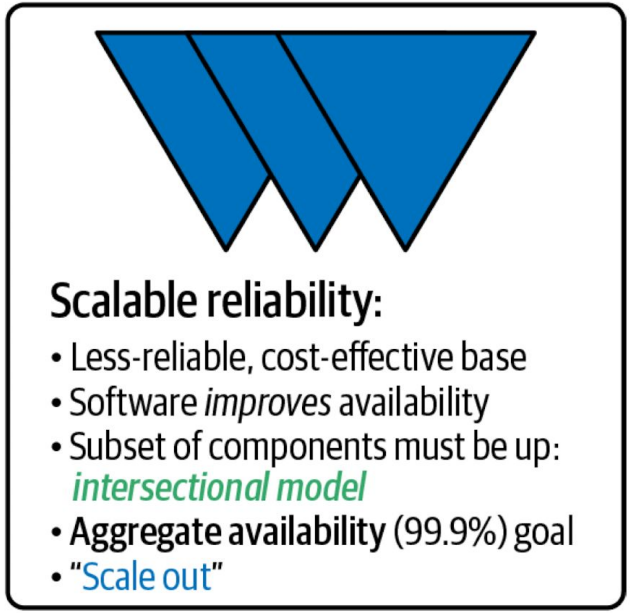
結論與摘要

From Scale up to Scale out



Component reliability:

- Solid base (big cold building, heavy iron, redundant disks/net/power)
- All components must be up as much as possible: *union model*
- **Total availability** as goal
- "Scale up"



Scalable reliability:

- Less-reliable, cost-effective base
- Software *improves* availability
- Subset of components must be up: *intersectional model*
- **Aggregate availability** (99.9%) goal
- "Scale out"

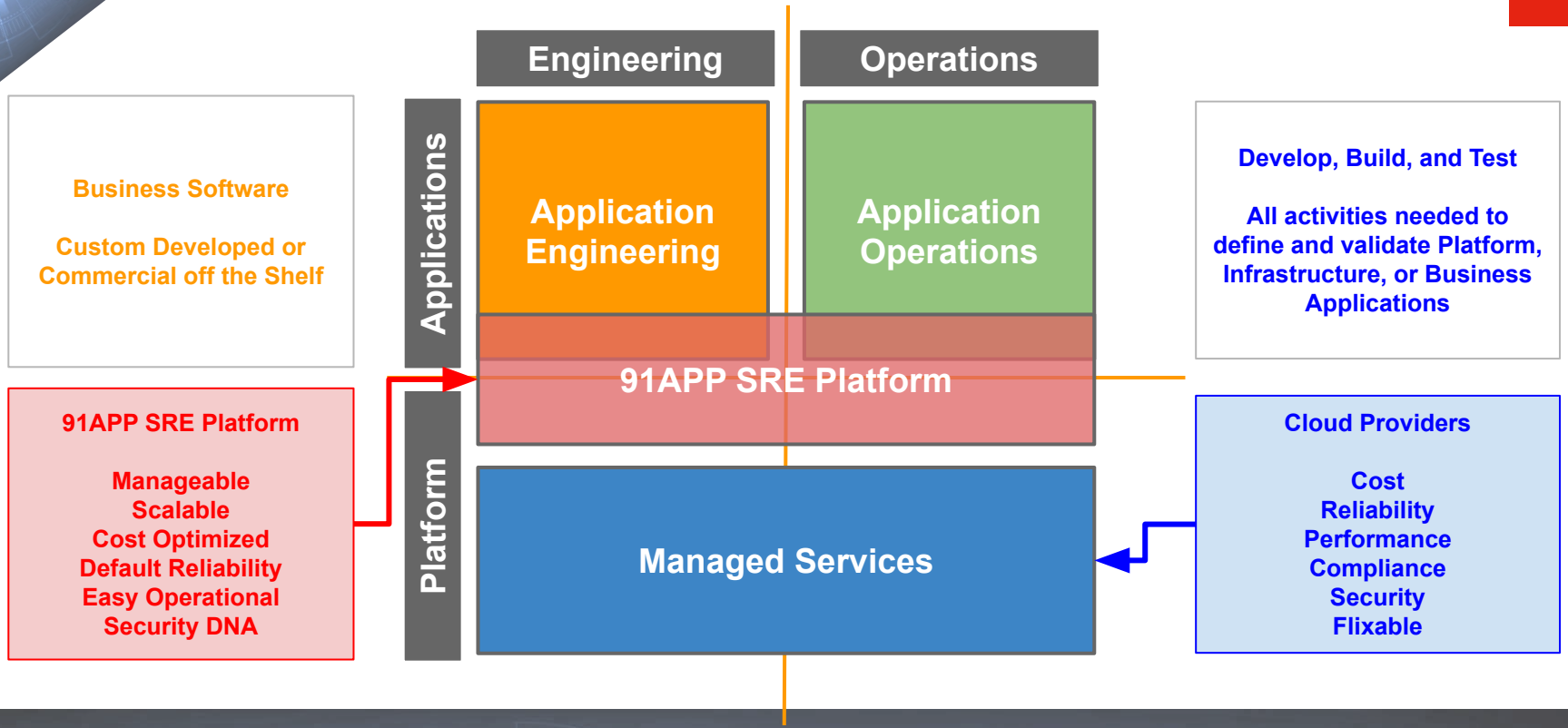
Source: Enterprise Roadmap to SRE: How to build and sustain an SRE Function
Figure 2-2. The pyramids of reliability.

公司的規模	SRE Team 的規模	SRE Team / Engineers / Leader
1 - 50	1 - 2	監控、事件管理、系統可靠度
50 - 100	1 - 5	系統可靠度、SLO、降低系統複雜度
100 - 300	1 - 10	DR、規範化、標準化、API
300 -	5 - 15	Cost、平台化、規模化

91APP

Source: AWS Well-Architected Framework - Operational Excellence Pillar

91APP SRE Platform



參考資料

1. Enterprise roadmap to SRE
2. AWS Well-Architected Framework - Operational Excellence Pillar

相關資料

1. 2017: 淺談系統監控與 CloudWatch 的應用 - AWS User Group Taiwan
2. 2018: 從緊急事件 談 SRE 應變能力的培養 - DevOpsDays Taipei 2018
3. 2018: Ops as Code using Serverless - iTHome
4. 2018: 邁向 API 經濟 - API Gateway 導入之旅 - AWS Summit 2018
5. 2019: 聊聊軟體交付的濫觴談產出物管理 - DevOps Meetup #22
6. 2018: 微服務的基礎建設 - Service Discovery, Andrew Wu - DevOpsDays Taipei 2018)
7. 2019: 從零開始的 Configuration Management - Levi Chen (DevOpsDay 2019)
8. 2020: 零售品牌經營「電商」該把資源投入在哪裡？- Rick Hwang
9. 2020: 災難演練 @ AWS 實戰分享 - AWS reInvent reCAP 2019
10. 2020: 在矩陣型組織裡，如何有效管理AWS 的成本結構與系統架構 - AWS Summit 2020

Site Reliability Engineering Taiwan

公開社團 · 4,527 位成員

已加入 ▾ ＋ 邀請

https://www.facebook.com/groups/sre.taiwan

91APP

Thank you!

免費建立 AWS 帳號
即可至攤位領取限量好禮

掃描 QR CODE 填寫問卷
領 AWS Credit 25美元

SRE Site Reliability Engineering
CONFERENCE 2022
04/29（五）｜富邦國際會議中心