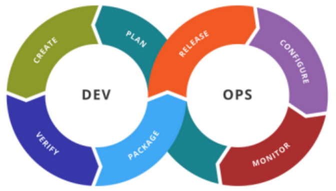
CloudnLoud

Community

Qlik

DATA ANALYTICS



LINUX FOUNDATION

INFOSEC Skills

TECH PLATFORM

Our Objective

- We want to building a passionated technical community wish to contribute for Childhood cancer.
- We also want a platform for the individuals want to share their knowledge for the benefit of the community.
- Mentoring the technical community members to grow in their career through helping them to present in conference, technical live events and webinars.

Community
CloudnLoud

Our Strength

- 4000 + Technology Meetups on Cloud & DevOps delivered across globe.
- Successfully delivered 640 corporate trainings and delivered 2000+ college trainings.
- Given career mentoring & Training to 1lakh + professionals in this 17 years

Community

CloudnLoud

Learner's

cloudnloud Tech Community

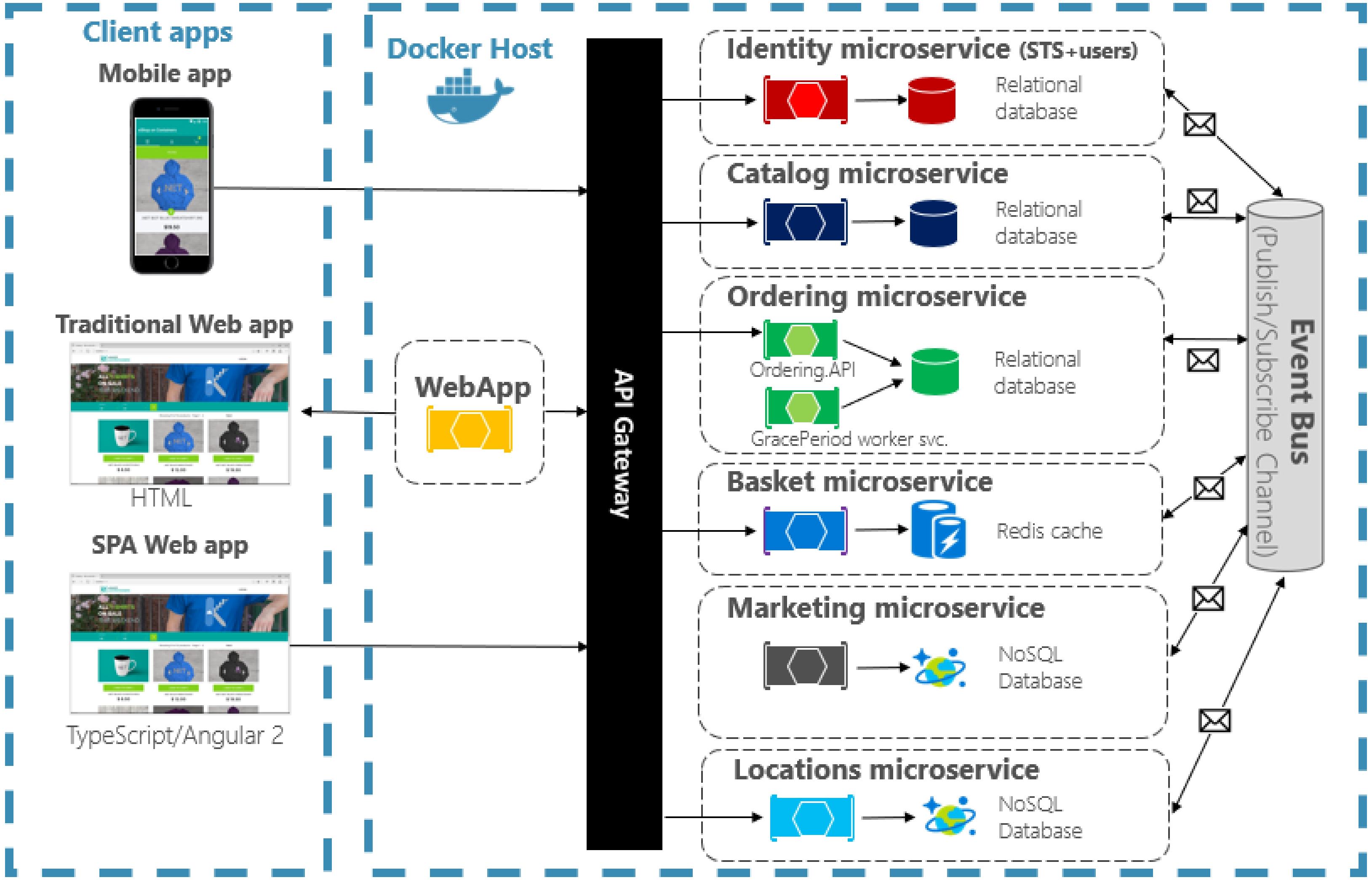
cloudnloud

cloudnloud

Community

CloudnLoud

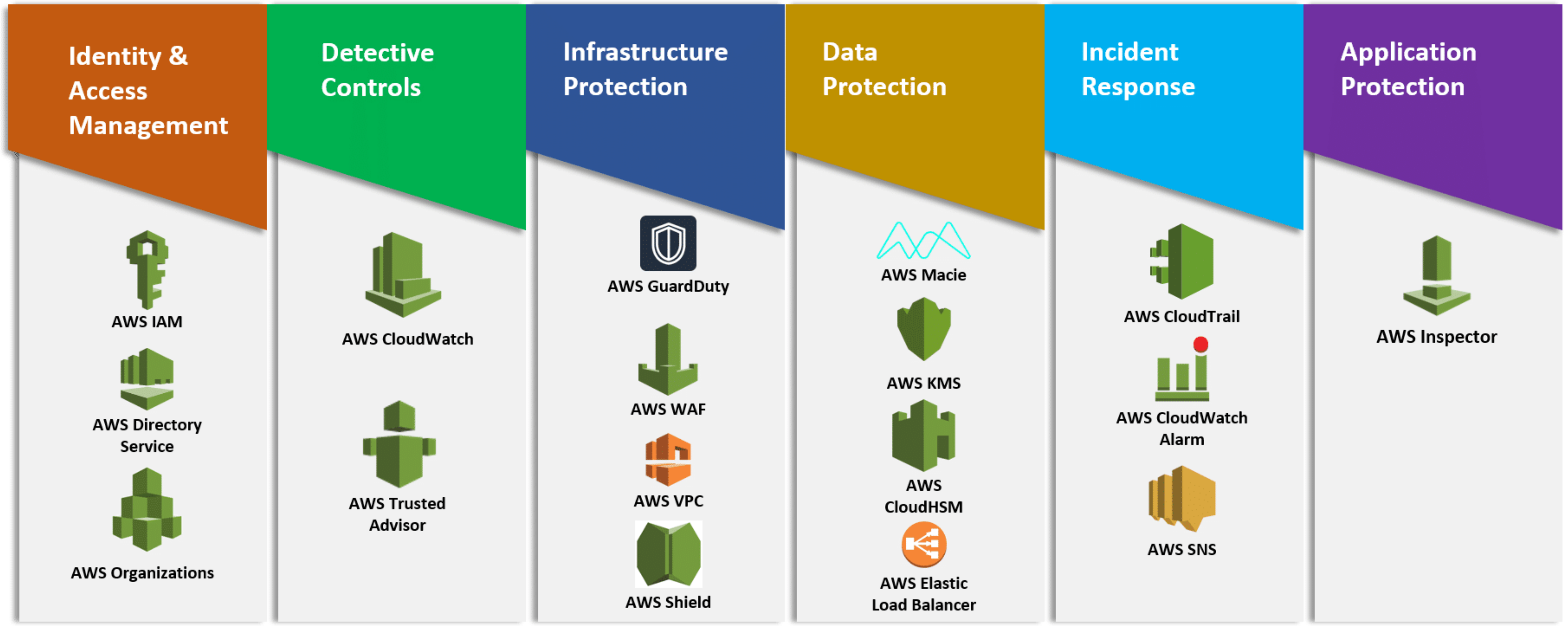
Azure Cloud Native Architecture



Community

CloudnLoud

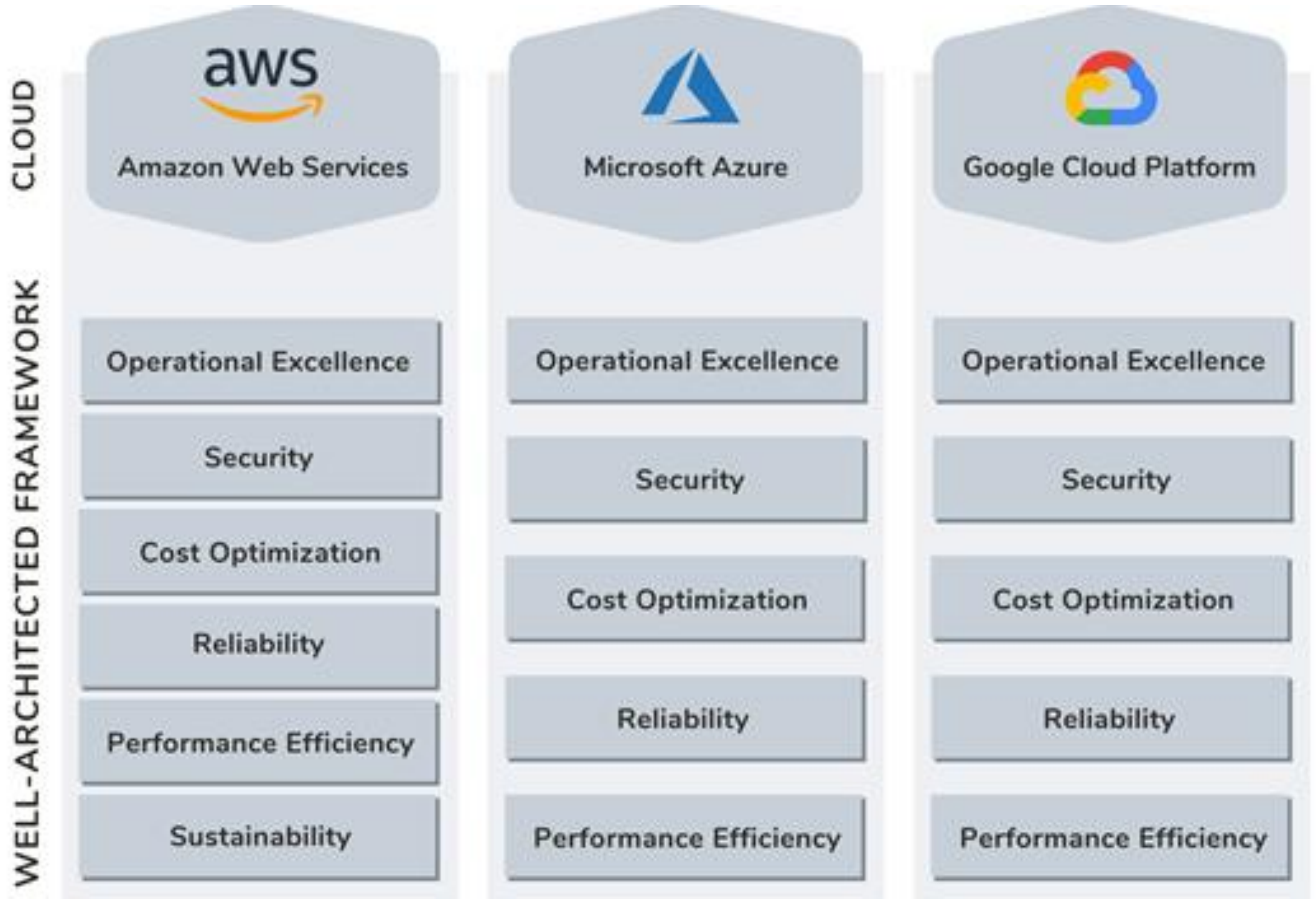
The 6 Pillars of the AWS Well-Architected Framework



Community

CloudnLoud

Pillars of the Cloud Well-Architected Framework



CLOUD	Amazon Web Services	Microsoft Azure	Google Cloud Platform
WELL-ARCHITECTED FRAMEWORK	Operational Excellence	Operational Excellence	Operational Excellence
	Security	Security	Security
	Cost Optimization	Cost Optimization	Cost Optimization
	Reliability	Reliability	Reliability
	Performance Efficiency	Performance Efficiency	Performance Efficiency
	Sustainability		

Community
CloudnLoud

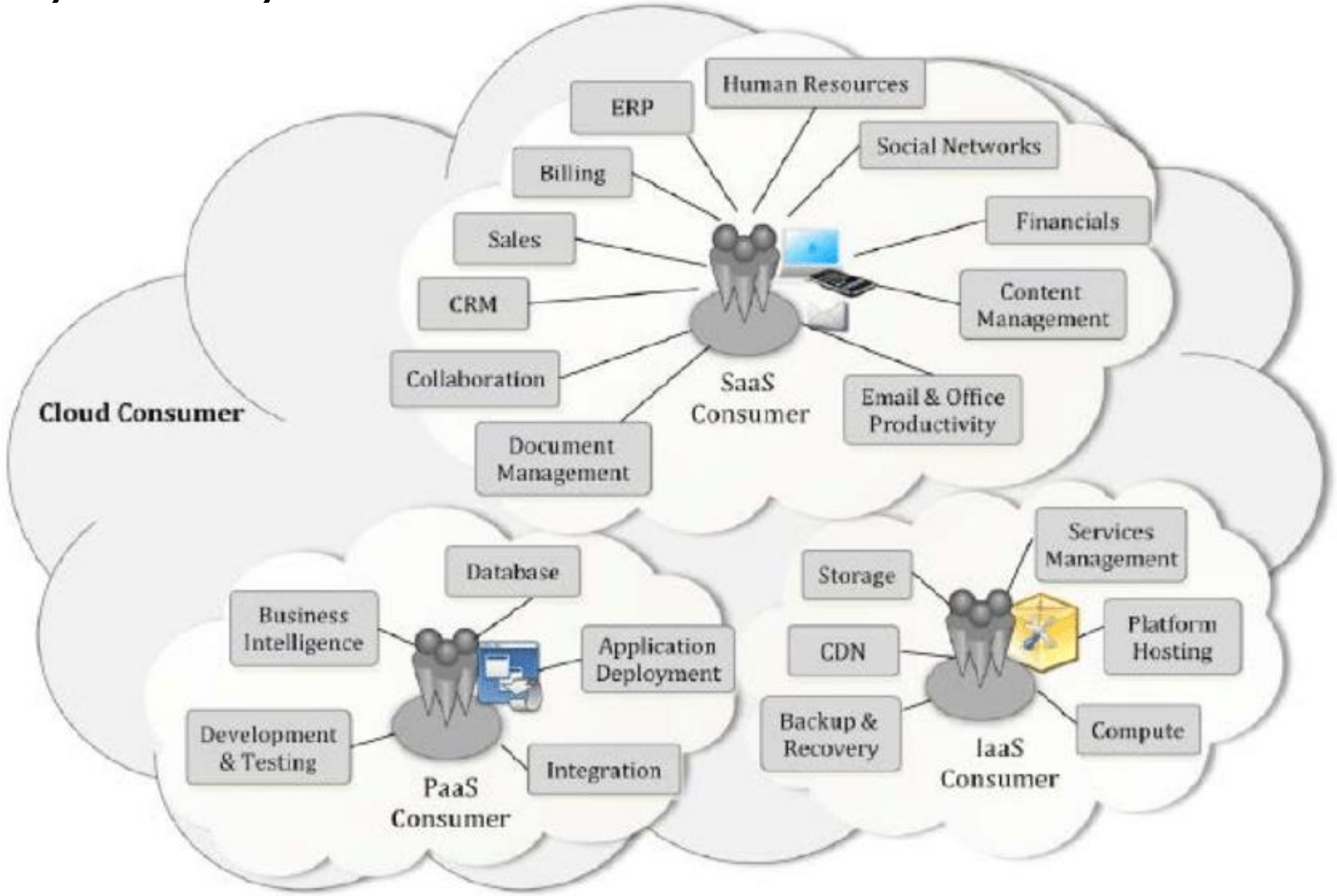
Pillars of the Cloud Well-Architected Framework

Cloud CORPS Landscape - Cloud architecture frameworks & related resources

Clouds	Amazon Web Services	Microsoft Azure	Google Cloud Platform
Architecture Frameworks	Well Architected Framework	Well Architected Framework	Cloud Architecture Framework
	Operational Excellence	Operational Excellence	Operational Excellence
	Security	Security	Security, Privacy, Compliance
	Reliability	Reliability	Reliability
	Performance Efficiency	Performance Efficiency	Performance & Cost Optimization
	Cost Optimization	Cost Optimization	
			System Design
Tools	Well Architected Tool	Well Architected Assessment	[Not Available Yet]
Add-Ons	Domain Specific Lenses	[Not Available Yet]	[Not Available Yet]
Related Tools	AWS Trusted Advisor	Azure Advisor	[Not Available Yet]
Related Frameworks	Cloud Adoption Framework	Cloud Adoption Framework	Cloud Adoption Framework

Community
CloudnLoud

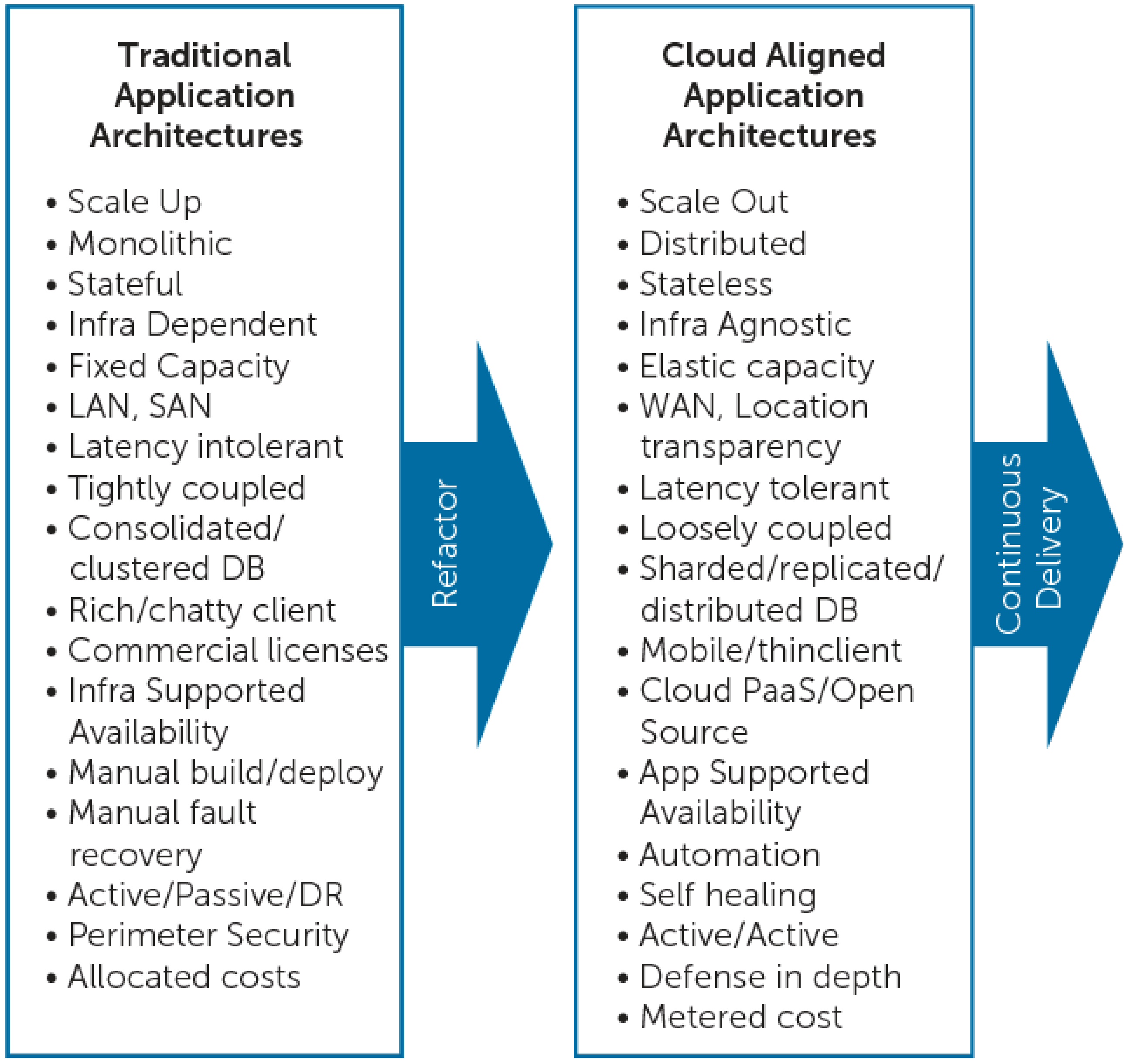
IAAS , PAAS , SAAS



Community

CloudnLoud

Traditional / Cloud Aligned



Traditional Application Architectures

- Scale Up
- Monolithic
- Stateful
- Infra Dependent
- Fixed Capacity
- LAN, SAN
- Latency intolerant
- Tightly coupled
- Consolidated/ clustered DB
- Rich/chatty client
- Commercial licenses
- Infra Supported Availability
- Manual build/deploy
- Manual fault recovery
- Active/Passive/DR
- Perimeter Security
- Allocated costs

The "Old World"

Cloud Aligned Application Architectures

- Scale Out
- Distributed
- Stateless
- Infra Agnostic
- Elastic capacity
- WAN, Location transparency
- Latency tolerant
- Loosely coupled
- Sharded/replicated/ distributed DB
- Mobile/thinclient
- Cloud PaaS/Open Source
- App Supported Availability
- Automation
- Self healing
- Active/Active
- Defense in depth
- Metered cost

The "New World"

Community
CloudnLoud

12-factor Methodology	Principle	Description
1	Codebase	The first principle is to maintain a single codebase for each application that can be used to deploy multiple instances/versions of the same app and track it using a central version control system such as Git.
2	Dependencies	As a best practice, define all the dependencies of the app, isolate them and package them within the app. Containerization helps here.
3	Configurations	Though the same code is deployed across multiple environments, configuration varies with the environment. As such, it is recommended to separate configurations from code and store them using environmental variables.
4	Backing Services	While using a backing service such as a database, treat it as an attached resource and define it in the configuration file so that you can replace the attached resource with a similar service by simply changing the configuration details.
5	Build, Release, Run	Build, Release and Run are the three important components of a software development project. The 12-factor methodology recommends that these three components should be separated and managed so as to avoid code breaks.
6	Processes	While the app contains multiple processes, it is important to run all the processes as a collection of stateless processes so that scaling becomes easy while unintended effects are eliminated. Each process does not need to know the state of other processes.
7	Port-Binding	Contrary to traditional web applications that are a collection of servlets and contain dependencies, 12-factor apps are free from run-time dependency. They listen on a port to make the services available to other apps. eg: Port 80 for web servers, port 22 for SSH, port 27017 for MongoDB, port 443 for HTTPS etc.

Community

CloudnLoud

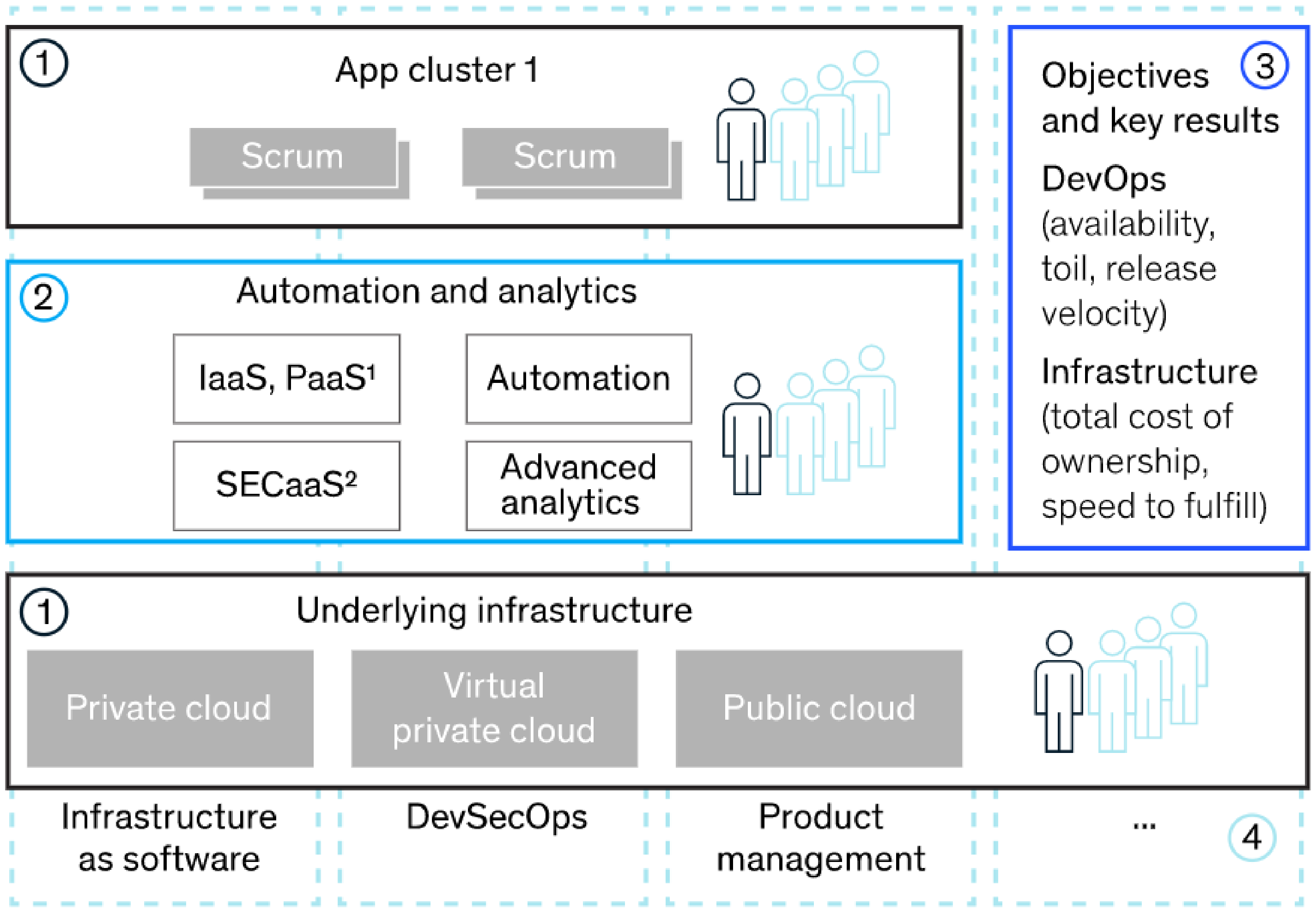
12-factor Methodology	Principle	Description
8	Concurrency	By running multiple instances simultaneously, you can manually as well as automatically scale applications based on predefined values. As dependencies are isolated in containers, apps can run side by side on a single host without causing any issues.
9	Disposability	When applications built on a cloud native application architecture go down, the app should gracefully dispose of broken resources and instantly replace them, ensuring a fast start up and shutdown. Being completely disposable, it gives the flexibility to start, stop or modify apps at the go.
10	Dev / Prod Parity	For applications to deliver consistent performance across different platforms, it is recommended to minimize differences between development and production environments. Building automated CI/CD pipelines, VCS, backing services and containerization will help you in this regard.
11	Logs	For better debugging, apps should create logs as event streams without worrying about where they are stored. Log storage should be decoupled from the app. The job of segregation and compilation of these logs lies on the execution environment.
12	Admin Processes	One-off tasks such as fixing bad records, migrating databases are also a part of the release. It is recommended to store these tasks in the same codebase

Community

CloudnLoud

AWS Pillers

https://cloudtweaks.com/2019/04/pillars-of-aws-well-architected-framework/


1
App cluster 1
Scrum
Scrum
2
Automation and analytics
IaaS, PaaS¹
Automation
SECaaS²
Advanced analytics
Objectives and key results
3
DevOps (availability, toil, release velocity)
Infrastructure (total cost of ownership, speed to fulfill)
1
Underlying infrastructure
Private cloud
Virtual private cloud
Public cloud
Infrastructure as software
DevSecOps
Product management
...
4

1. **DevOps/site reliability engineering**
   Small teams responsible for resiliency, "toil" reduction, and DevSecOps adoption

2. **Productized infrastructure services**
   Cross-functional teams responsible for building and maintaining discrete products

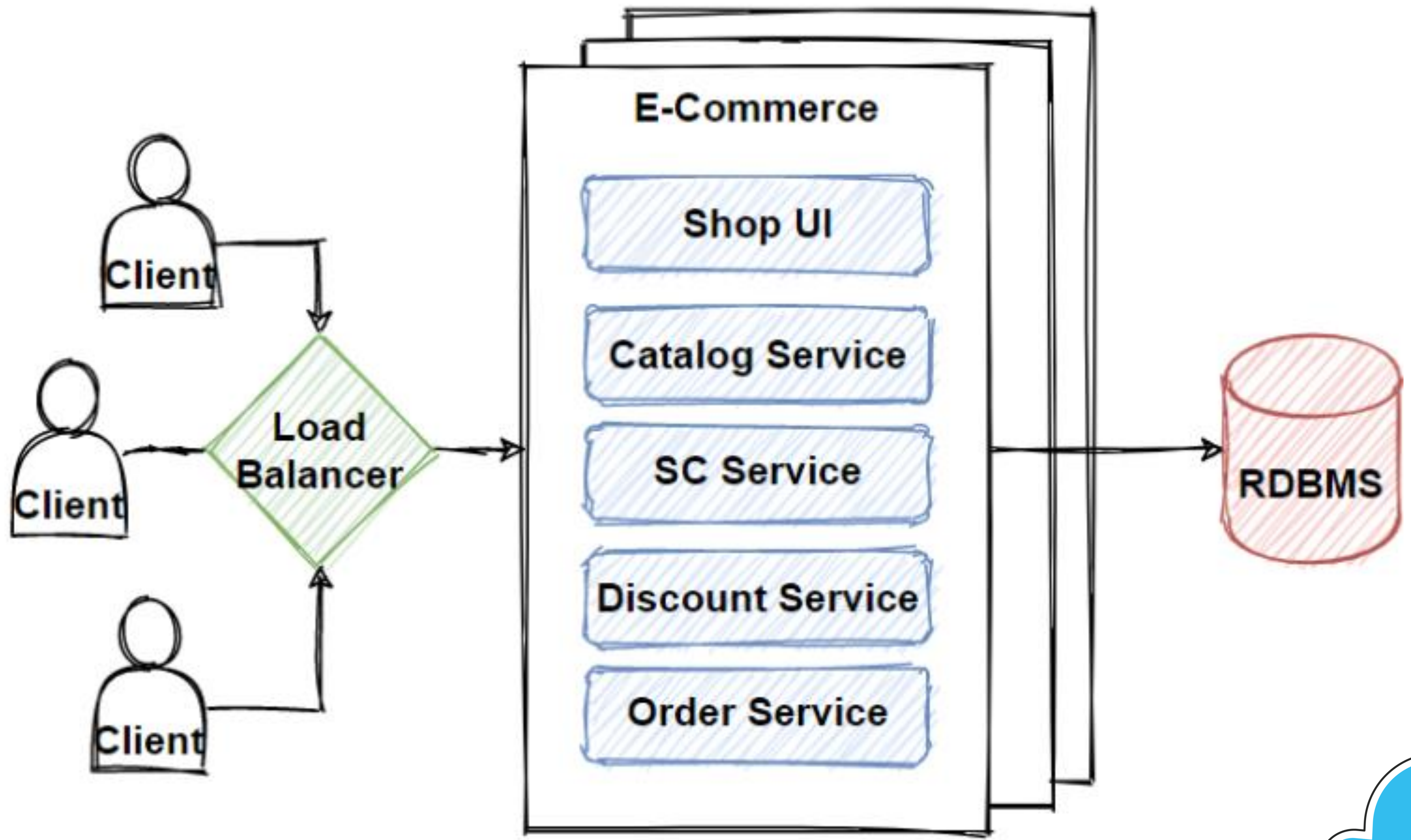
3. **Outcome-driven governance**
   Delivered through agile ways of working

4. **Engineering-centric capabilities**
   Focused on building world-class engineering talent and minimizing low-value operations

[1] IaaS = infrastructure as a service; PaaS = platform as a service.
[2] SECaaS = security as a service.

CloudnLoud Community

Monolithic Architecture

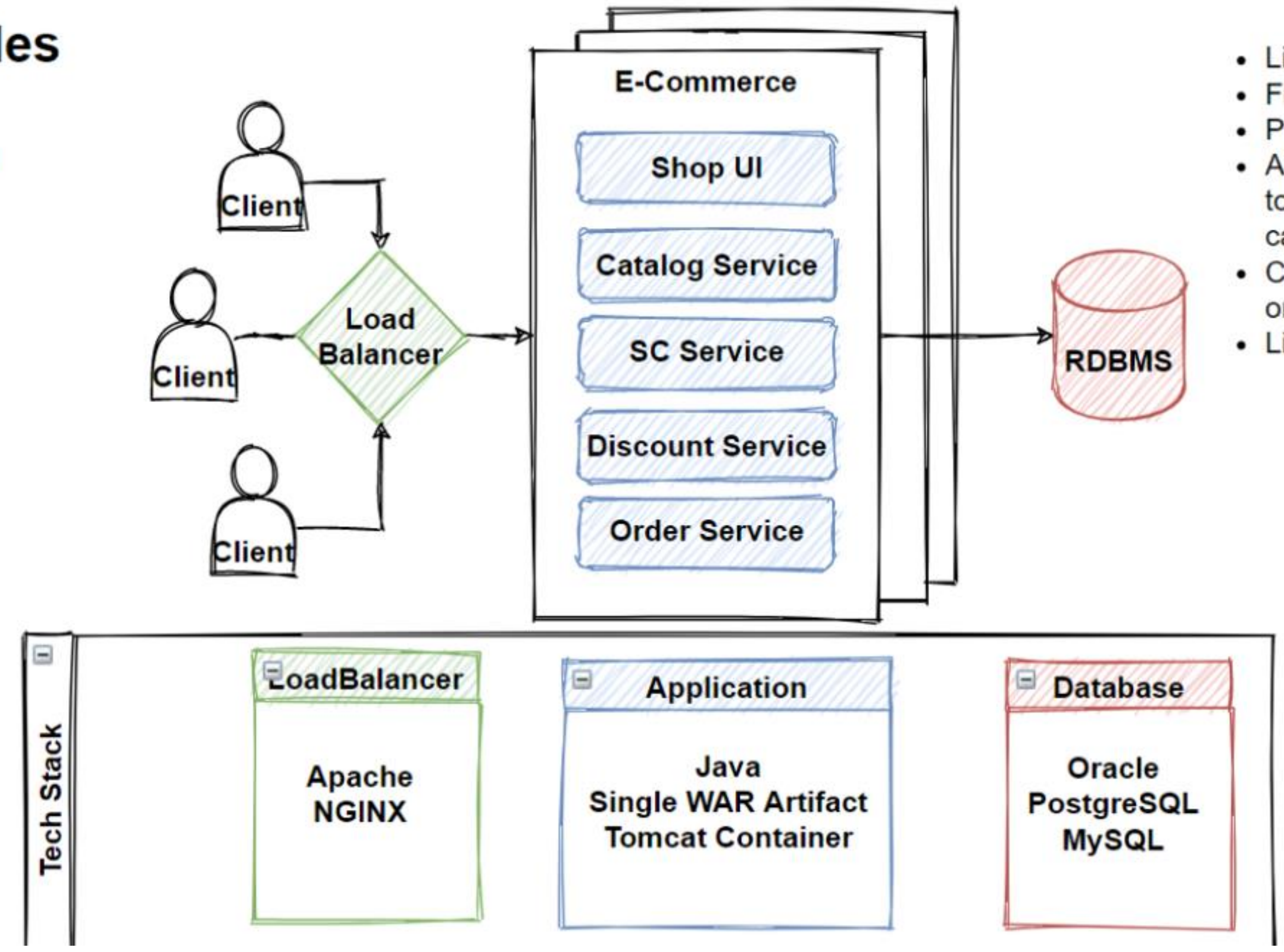


Community
CloudnLoud

Monolithic Architecture

Principles

- KISS
- YAGNI



Functional Requirements

- List products
- Filter products as per brand and categories
- Put products into the shopping cart
- Apply coupon for discounts and see the total cost all for all of the items in shopping cart
- Checkout the shopping cart and create an order
- List my old orders and order items history

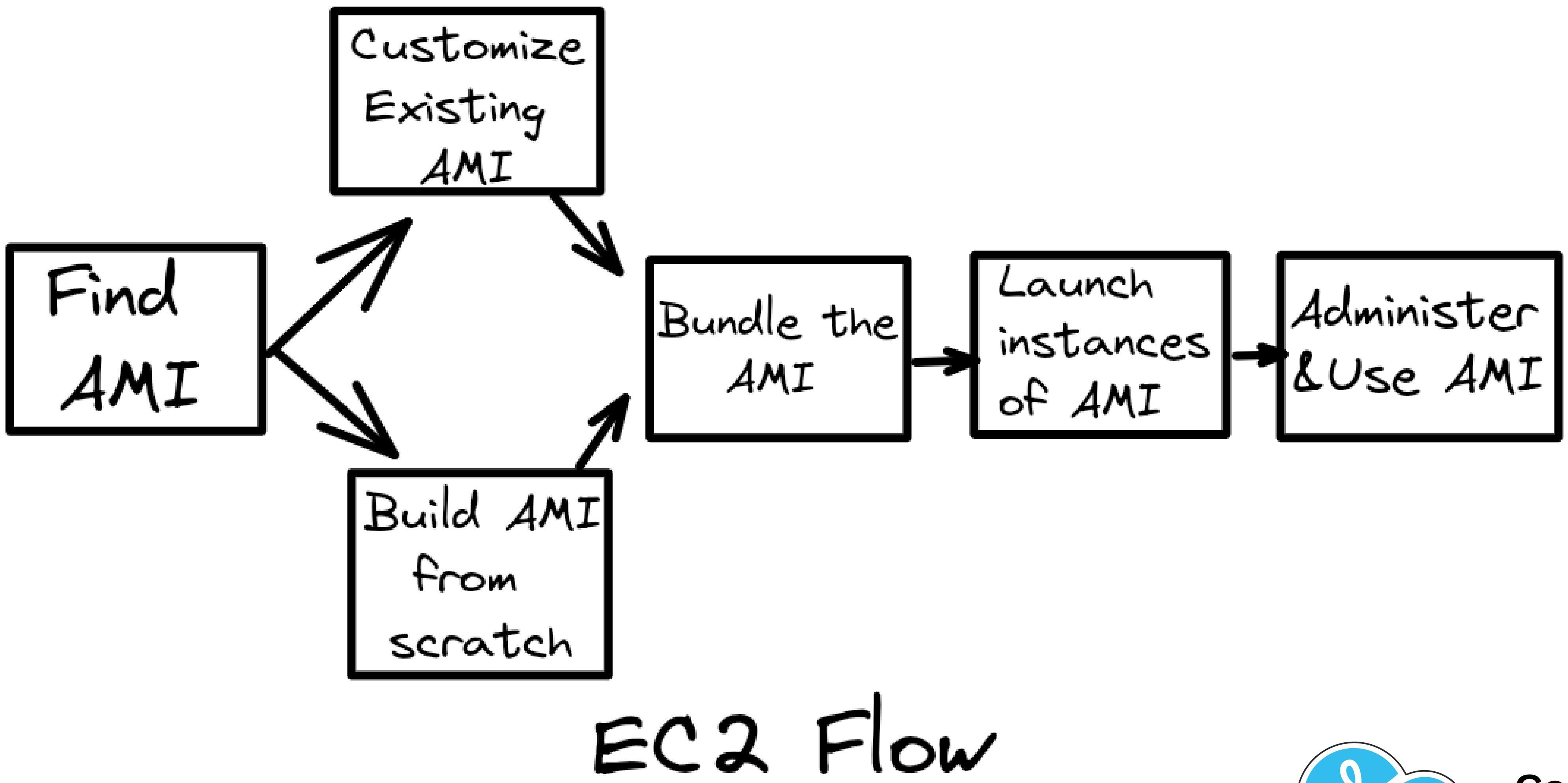
Non-Functional Requirements

- Scalibility
- Increase Concurrent User

Community

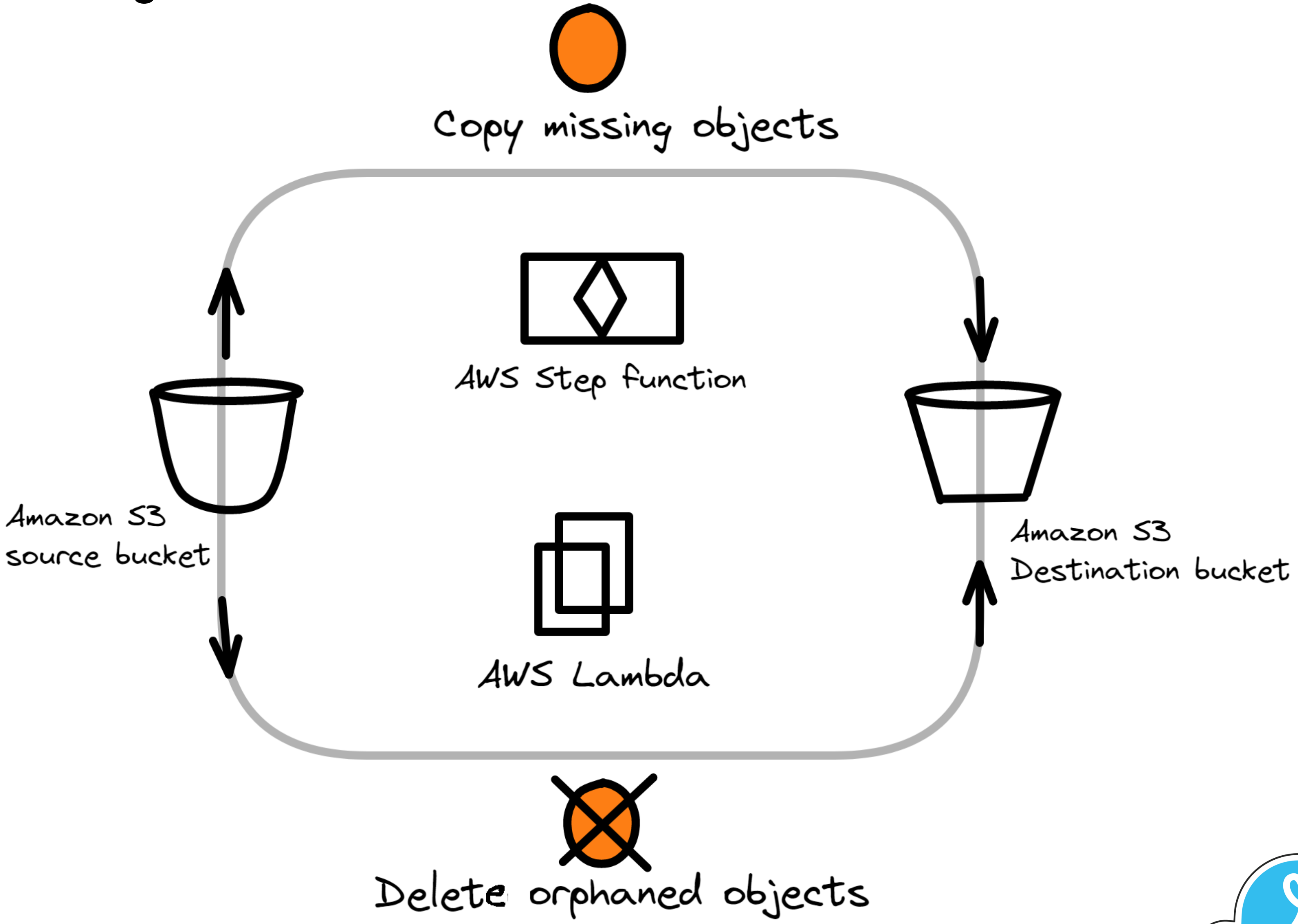
CloudnLoud

Amazon Elastic Compute



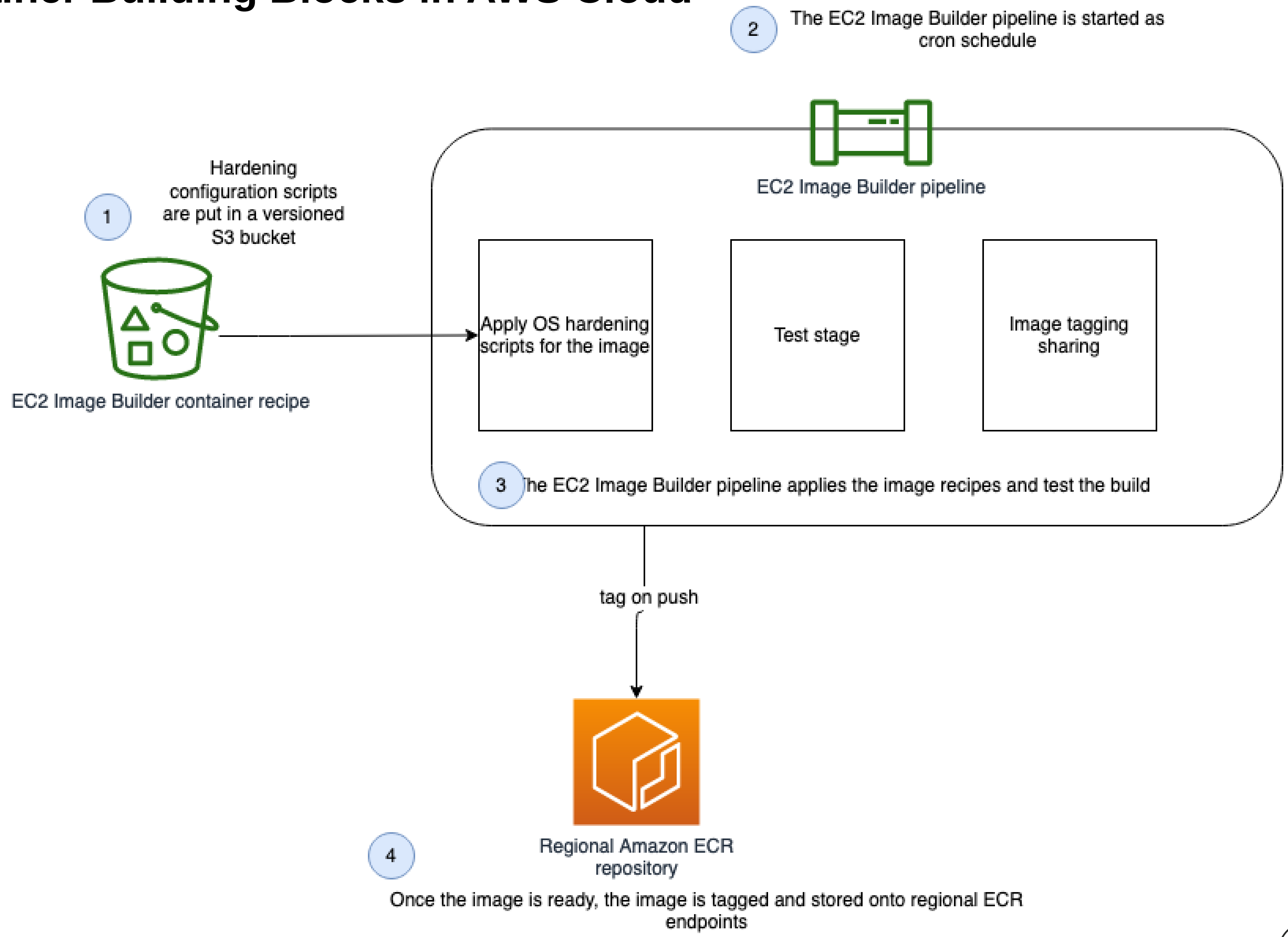
CloudnLoud Community

Simple Storage Service



Community
CloudnLoud

Container Building Blocks in AWS Cloud


2
The EC2 Image Builder pipeline is started as cron schedule
1
Hardening configuration scripts are put in a versioned S3 bucket
EC2 Image Builder pipeline
EC2 Image Builder container recipe
Apply OS hardening scripts for the image
Test stage
Image tagging sharing
3
The EC2 Image Builder pipeline applies the image recipes and test the build
tag on push
4
Regional Amazon ECR repository
Once the image is ready, the image is tagged and stored onto regional ECR endpoints

Community

CloudnLoud

Our Contact Details

Address:

Online

WhatsApp Only Number:

+91 8939984529

Email Address:

info@cloudnloud.com

Community

aws

Qlik

DATA ANALYTICS

DEV OPS

CREATE PLAN VERIFY PACKAGE RELEASE CONFIGURE MONITOR

LINUX FOUNDATION

INFOSEC Skills

CloudnLoud

TECH PLATFORM