KusionStack Origin, present and future

KusionStack Team

Agenda

01 Origin

02 Goal

03 Solution

04 Tech

05 Practice

06 Future

Origin

Cloud-native technologies

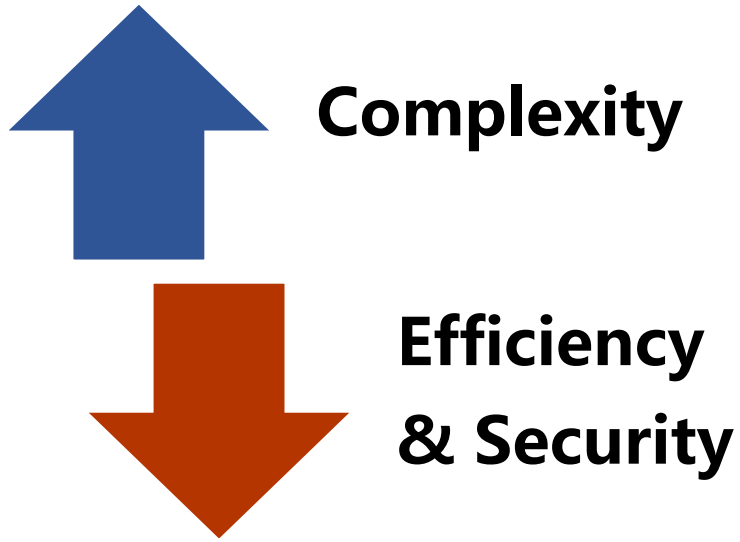
Are eating the world

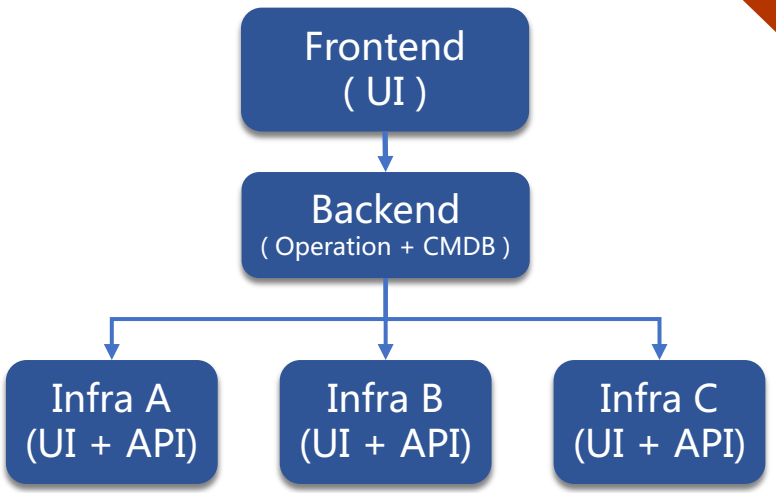
- First-party approach
- Hybrid cloud, Multi cloud
- Hybrid technology solutions
- DevOps, Self-Service
- Abstraction, management, devex

Diversity, scale and change

Create ongoing challenges

- Classic Ops platform: insufficient **openness**, **flexibility, security** and **scalability**
- Community DevOps tool: don't meet '**enterprise-grade**' needs





Classic PaaS is no longer applicable

Goal

Effective Teamwork

Enable overall success

Dev and Ops Silos

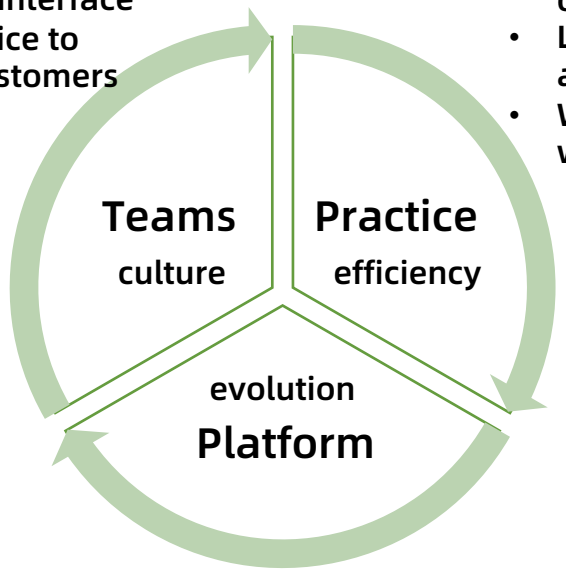
DevOps Team Silo

Dev Don't Need Ops

Collaborate, Automate

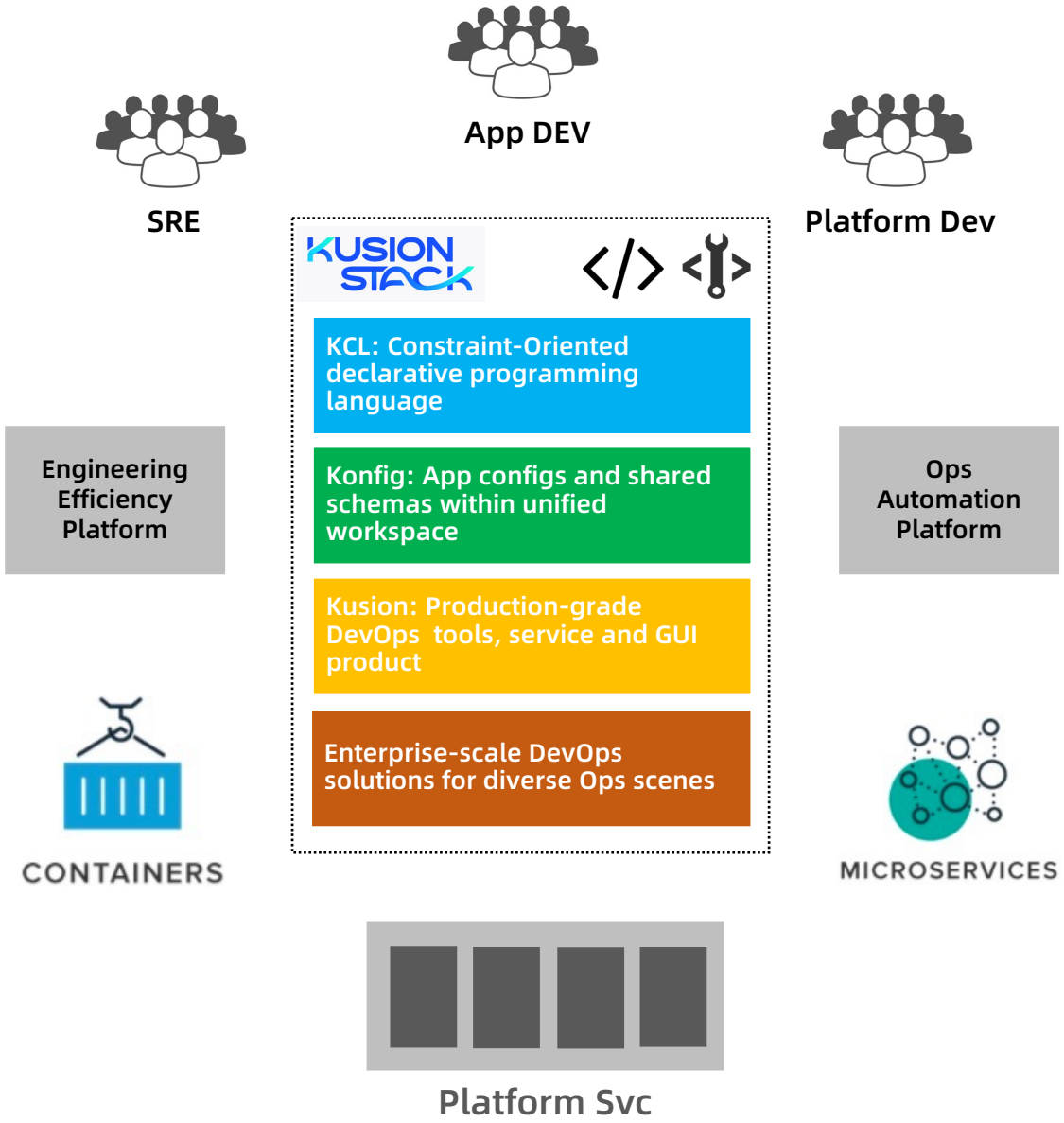
Make scaled DevOps possible

- Collaborate and share across teams
- One-stop working space and interface
- Better service to internal customers



- Codify
- Efficient Ops business development
- Manage change based on commit
- Left-shifted inspection and analysis
- Weakening the process with practice

- Highly open CI/CD/CDRA platform
- Unified and single-source ‘fact’ management mechanism
- Extensible to all ops scenarios
- Continue to face new challenges at enterprise scale



A Stack to Deliver Value

Make scaled delivery agile

Collaborative Declarative

{Platform, App} Developers
Programmable

APP
- Dev
- SRE

PLATFORM
- Dev
- SRE

App Centric Shipping

App Ops Lifetime
Multi-{Tenant, Env}

Collaboration
Automation

Declarative
Dev-friendly workflow

KUSION STACK

App Config
- Instrumentation
- Security
- Runtime
- Deliverable

Defined by KCL

- GUI
- CI (GitHub Actions, GitLab CI)
- CD (argo, flux)
- CLI

Kusion
- Preview
- State
- Provision
- Watch
- Orchestration
- Transaction
- Kube Client
- TF Client

Codify Stack Engineering

Multi-Runtimes
Multi-Clouds

Cloud-Native App Runtimes

SOFASTACK, OCEANBASE, Prometheus, Istio, cilium, kubernetes

AWS

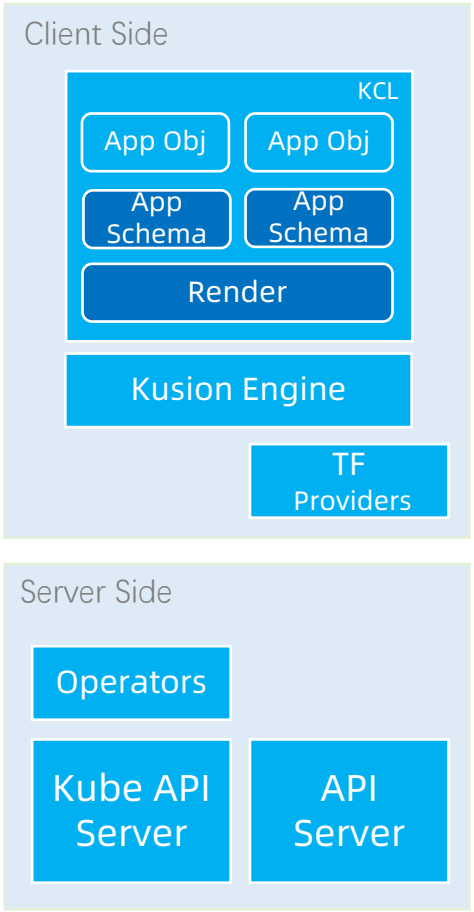
Organize | Code | Run

Solution

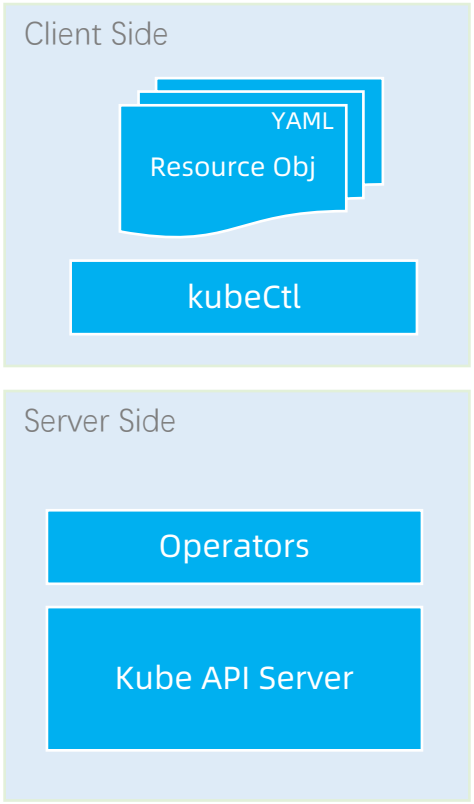
Kube Ops with X

Portable client solution with app centric interface

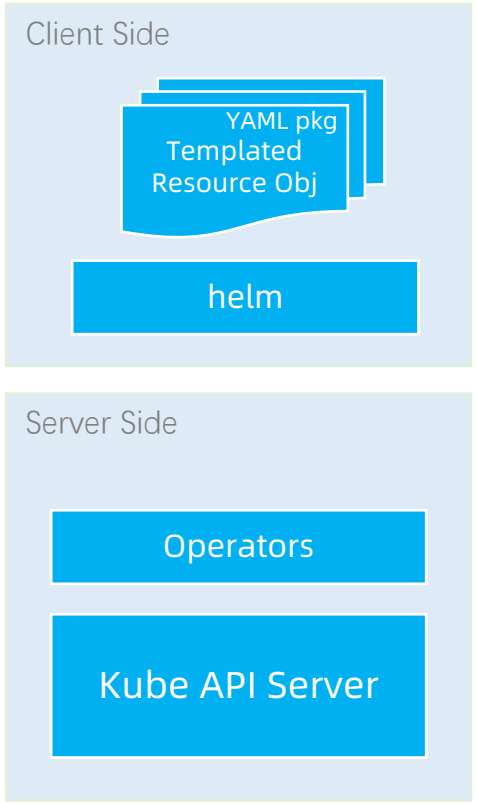
- **App Centric**
  - Modeling
  - Abstraction
  - Customization
  - Combination
  - Policy
- **Pure Client Solution**
  - Codify
  - Lightweight
  - Flexible
  - Scalable
  - Portable
  - Left-shifted stability
- **Hybrid-platform**
  - On Kube & TF
  - On Multi-Clouds
  - Provision
  - Orchestration
  - Visualization
  - ...
- **E2E Support**
  - Dev
  - CI
  - CD



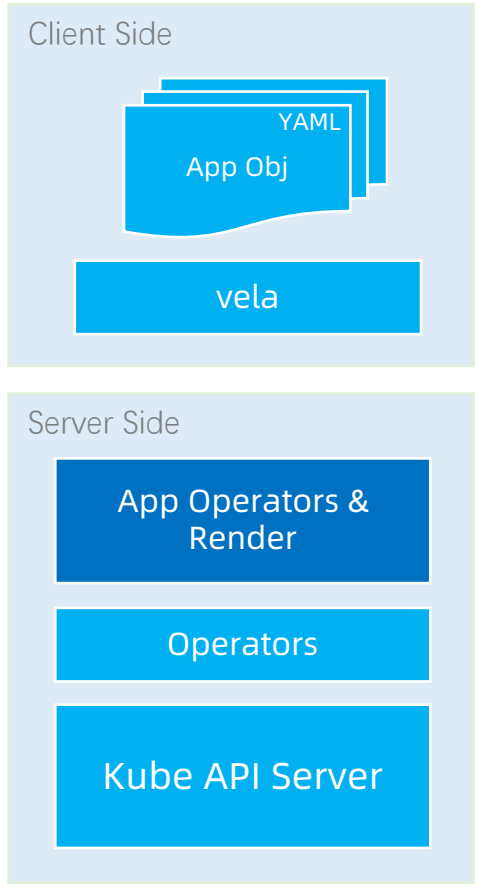
➢ **KusionStack**



➢ **Typical tool: Kustomize**



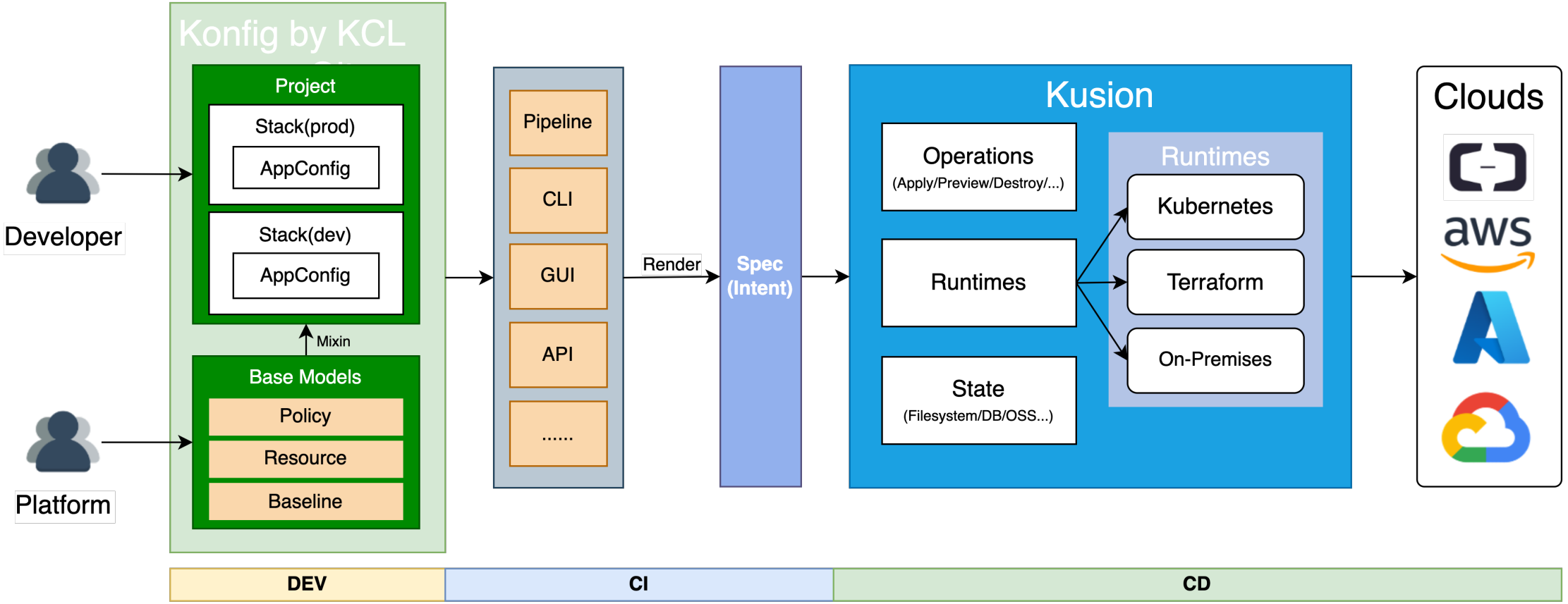
➢ **Typical tool: Helm**

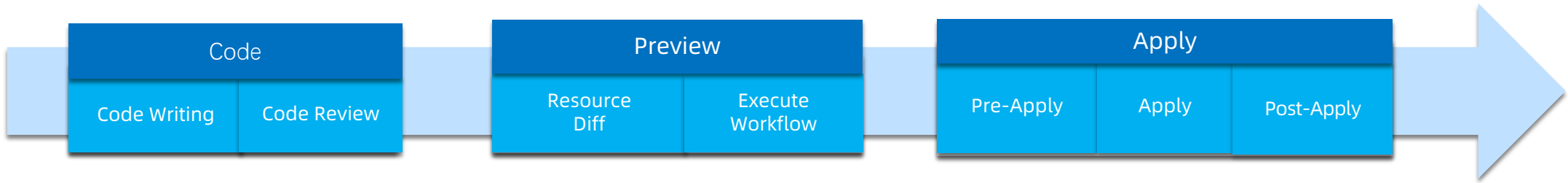


➢ **Typical tool: KubeVela**

Enterprise

Fast and reliable development and automation

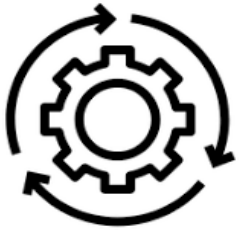




Automation

Make the greater flexibility and agility

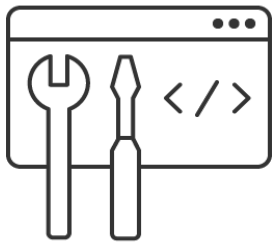
Portal



CI Suite

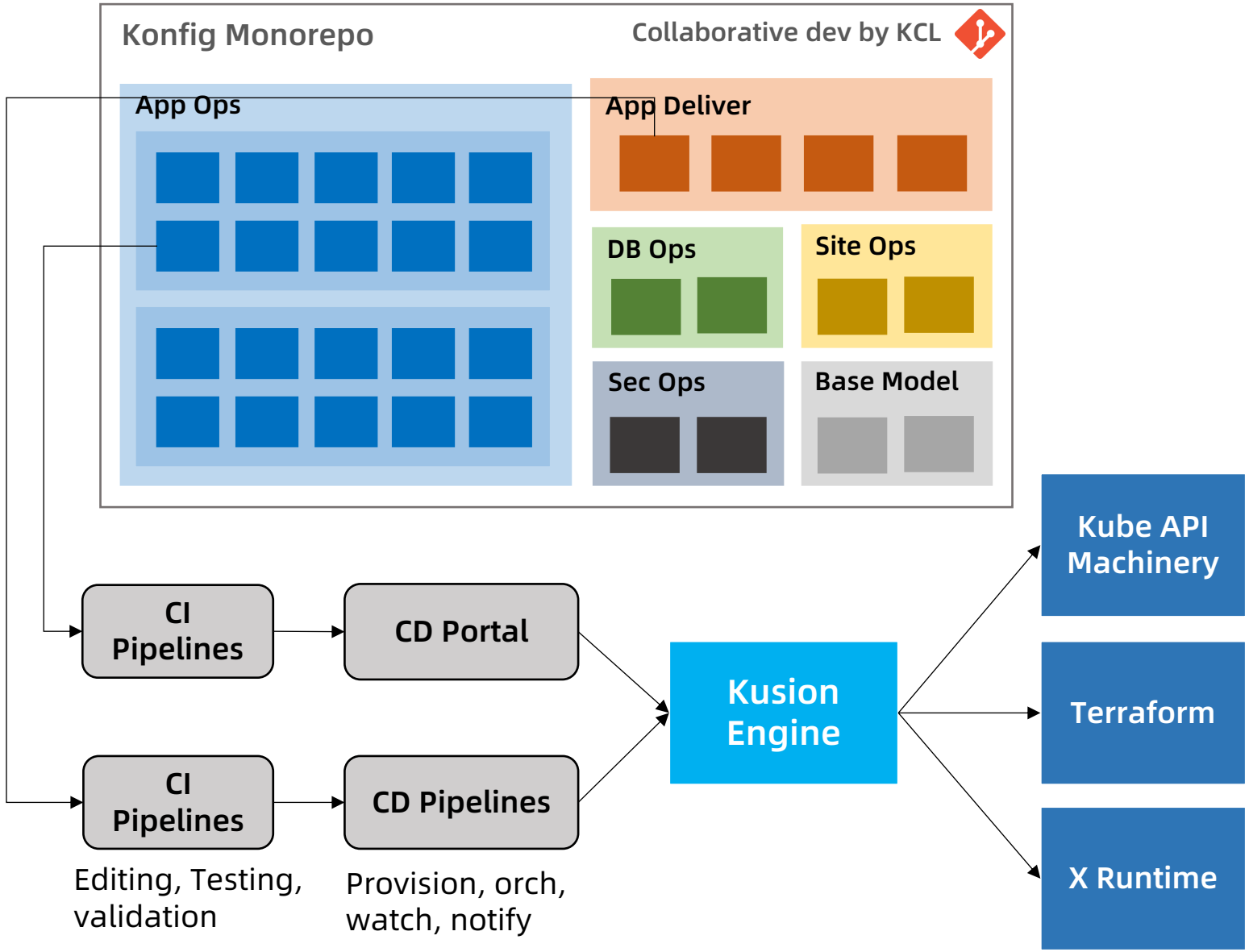
API

Tool



Code

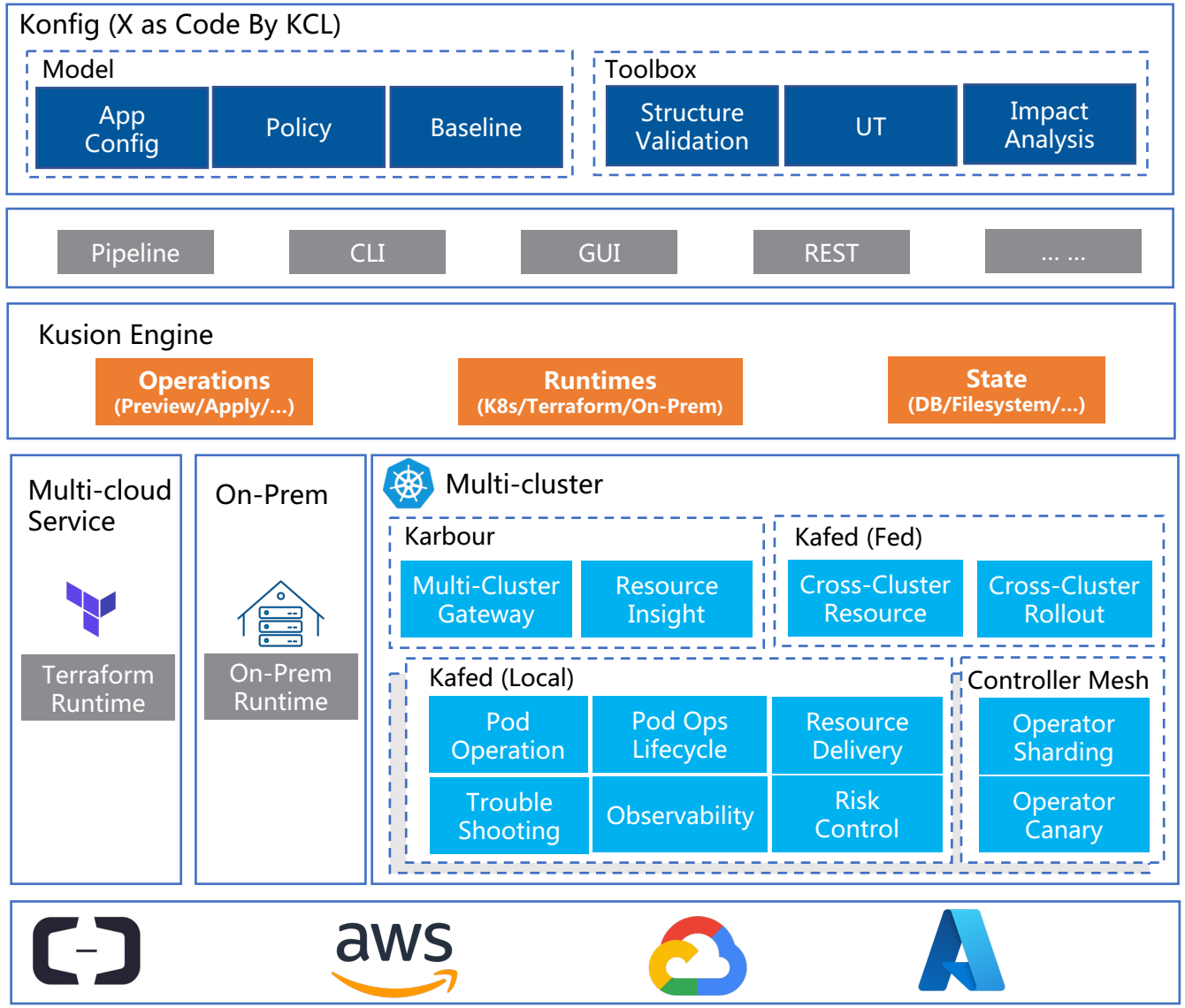
Doc



Tech

Arch

Build your IDP effectively and safely



Konfig & Kusion

An abstraction and management layer to deliver modern app

Monorepo

Organize all app confs in one repo with scalable project & stack structure

Vendor Agnostic

Write once, deliver any runtime, any cloud through a consistent workflow

Multiple Hierarchies

Natively support multi-tenant and multi-environment configuration

Lifecycle Management

Manage app from the first code to production-ready across multi-phases

Extendable Models

Extendable and reusable modeling by schema, mixin and other KCL mechanisms

Hybrid Resource

Orchestrate resources on various runtime in a managed manner

Kusion Engine

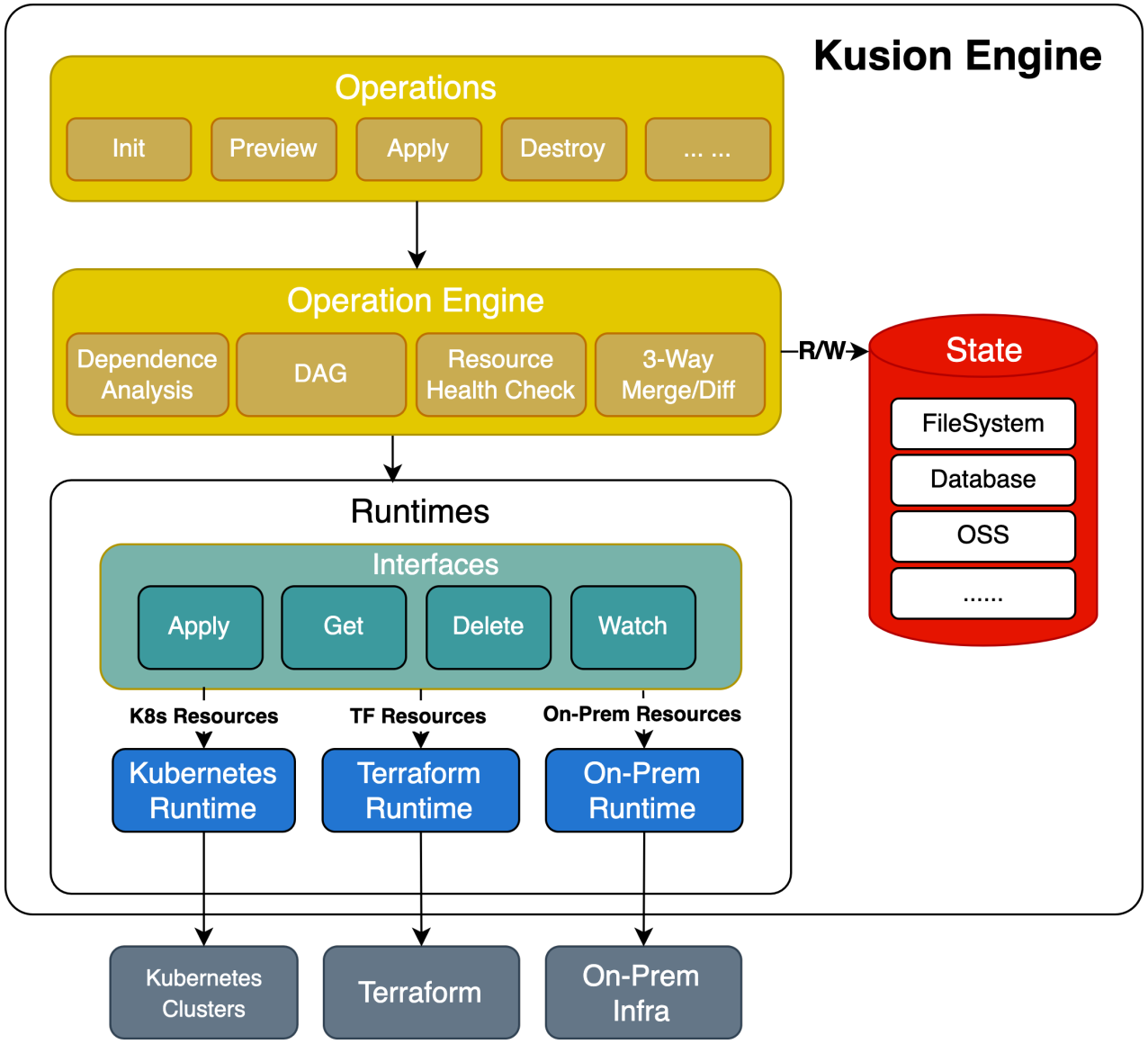
Managed resource across multiple runtimes

Kusion Engine: Platform engineering engine, responsible for all operations

Operations: Provide core capabilities such as resource management, orchestration, and live-diff for all Kusion operations commands.

Runtimes: represents infrastructures managed by Kusion, which interacts with heterogeneous infrastructure through a unified interface

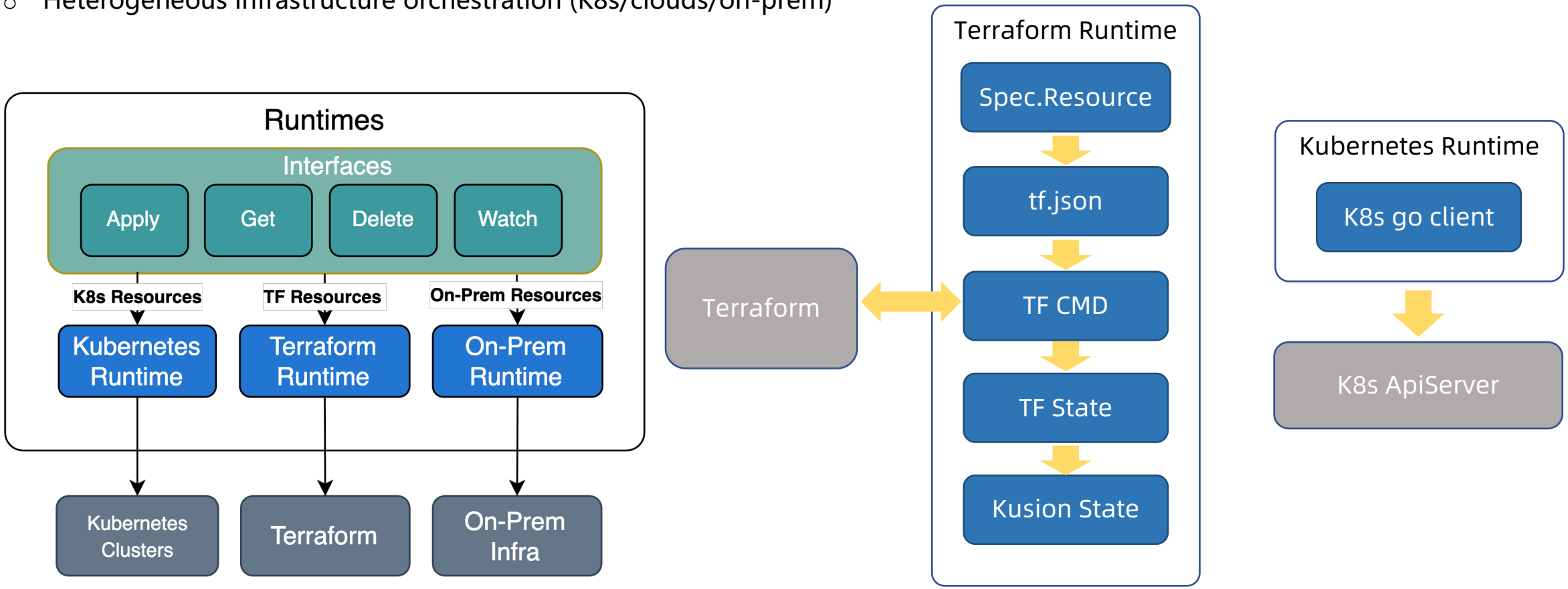
State: The mapping of real resources to Kusion used to resource management



Kusion Engine

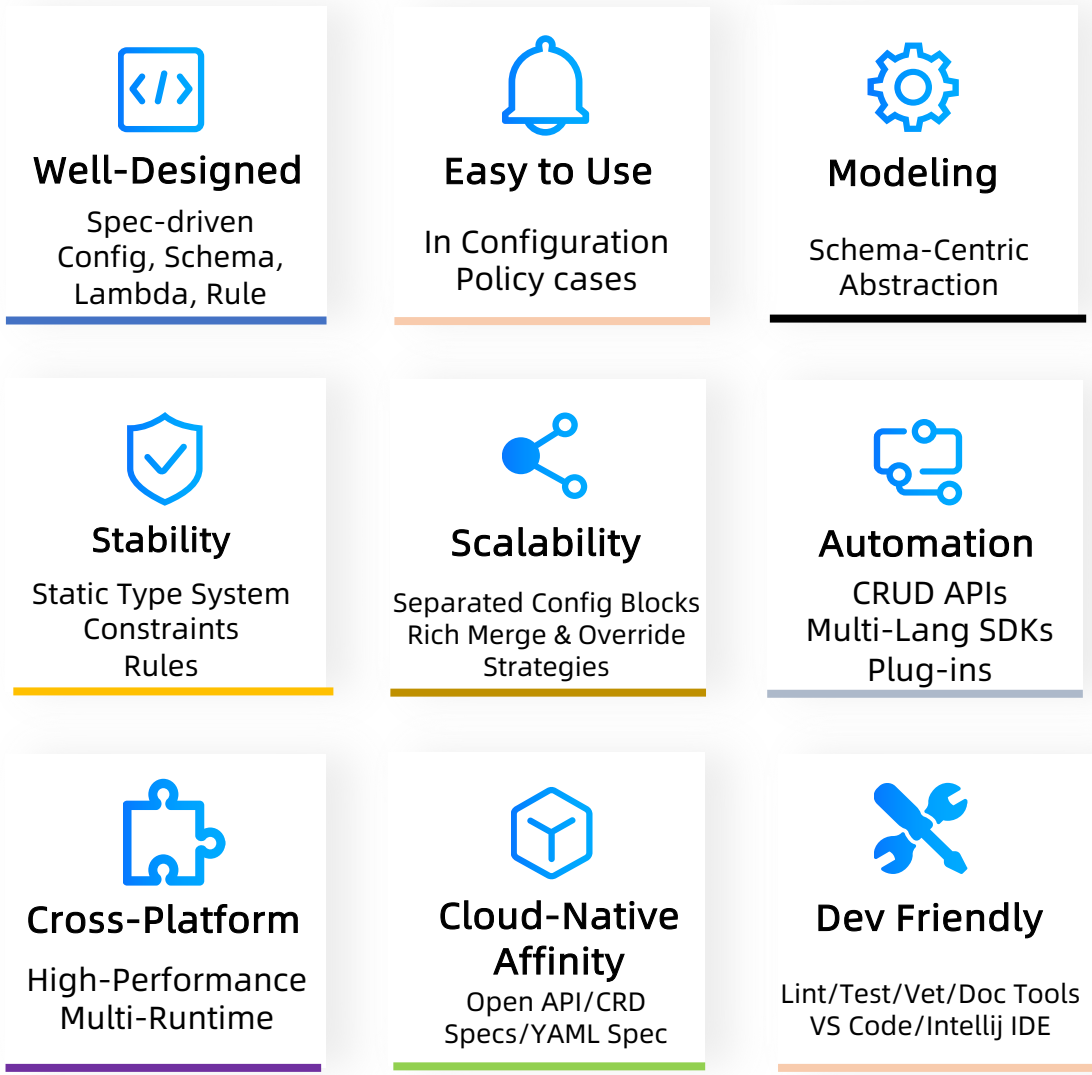
Enable heterogeneous infrastructure management

- Unified Management Interfaces
- Heterogeneous infrastructure orchestration (K8s/clouds/on-prem)



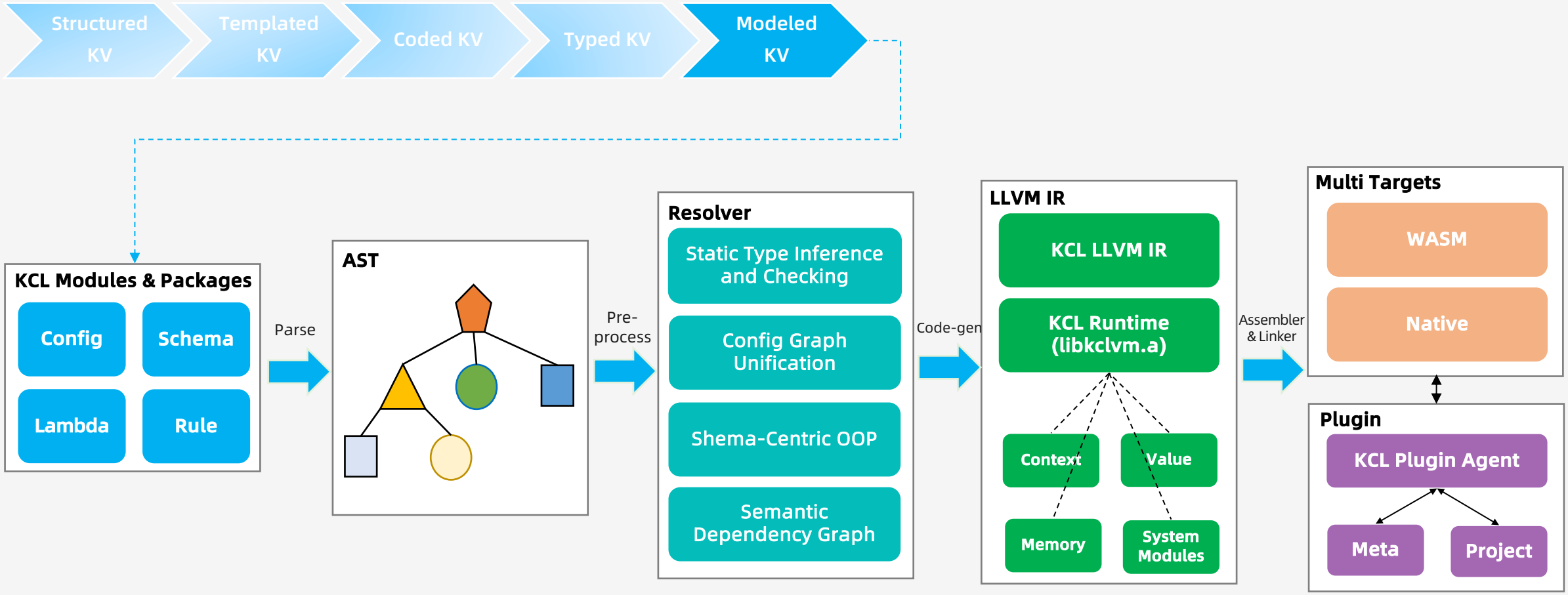
KCL

An Open Source Constraint-Based Record & Functional Language



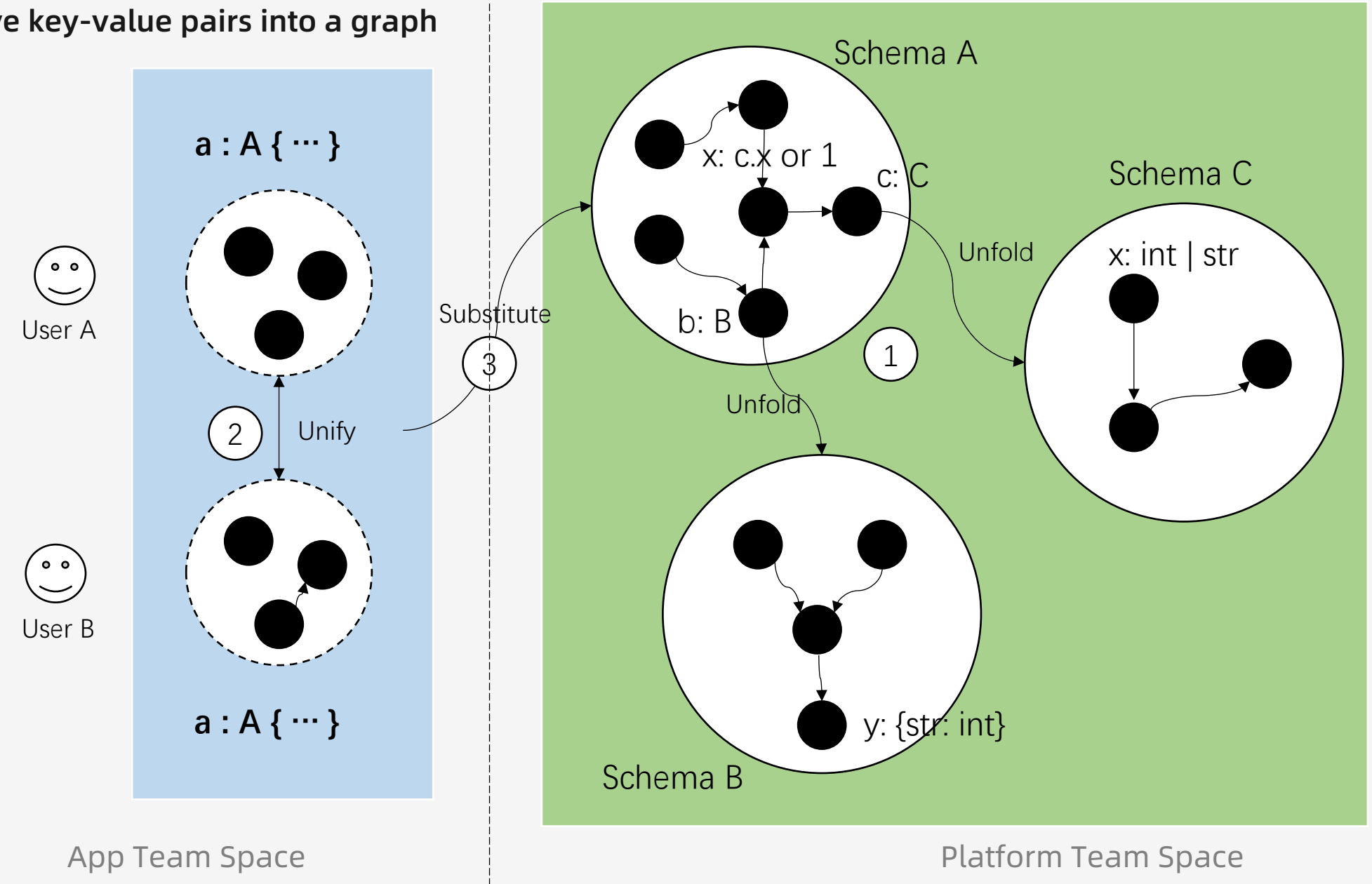
KCL

Config, Schema, Lambda, Rule



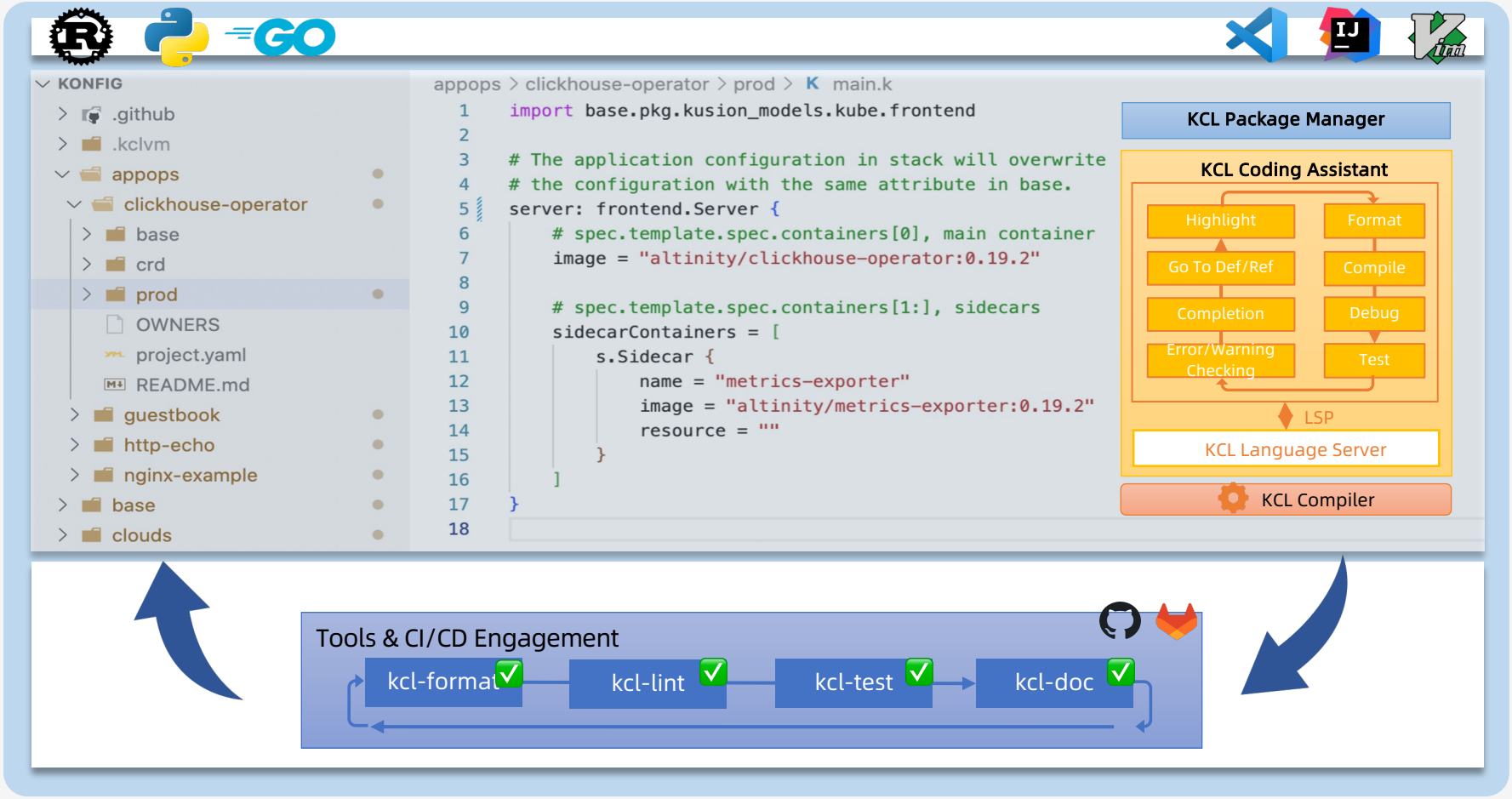
KCL Graph Model

Weave key-value pairs into a graph



KCL Tools & IDE Workspace

Make Ops collaborative coding and work happy



Practice

User Roles of Kusionized DevOps

App Dev

Roles
- End user

Goals
- Deliver and ops my app easier
- On any desired env and cloud

Favors
- Implicit and app-oriented working interface and process above infrastructure details
- Minimal investment in learning and practice in infrastructure and operation details

Pain points
- Too many fragmented technologies, processes and user interfaces in deliver and ops
- Too many infrastructure-oriented details to learn
- Growing cloud platforms to use

SRE

Roles
- Enabler
- End user

Goals
- Keep infra and ops stable, measurable and manageable
- Help & enable end users

Favors
- Participate directly in the work of platform design and construction to make the infrastructure more reliable and easy-to-use for app developers
- Deliver and manage apps that require high stability through easy-to-use tech and tools

Pain points
- Unable to directly participate in the construction of the platform
- Platform capabilities related to stability cannot be used by app developers faster

Platform Dev

Roles
- Provider & Enabler
- End user

Goals
- Deliver platform projects to multi-clouds
- Enable user-side self-service and reduce ops and service costs

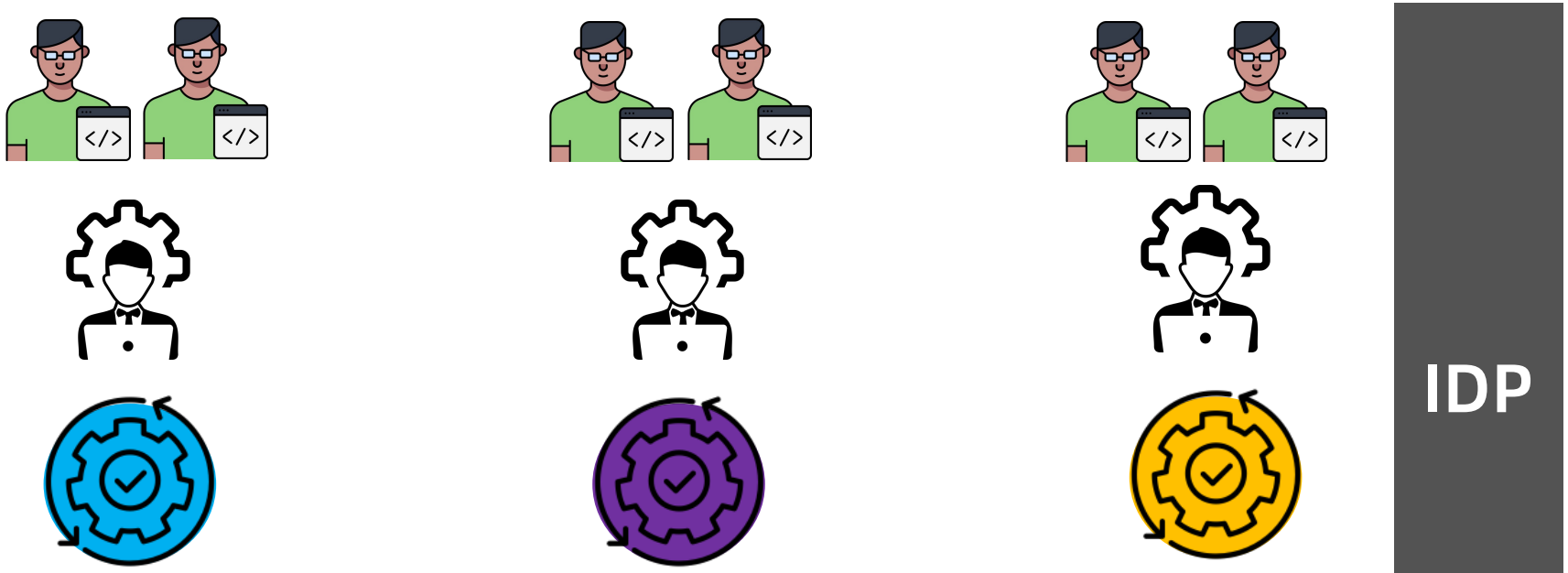
Favors
- Application developers can use platform capabilities in a self-service way
- Deliver platform apps using lightweight and open-source tech and tools in an explicit way

Pain points
- Unable to invest more time in R&D due to user supporting
- Unable to make app developers to access platform capabilities in a uniform, stable and low-cost way

Dev

Users

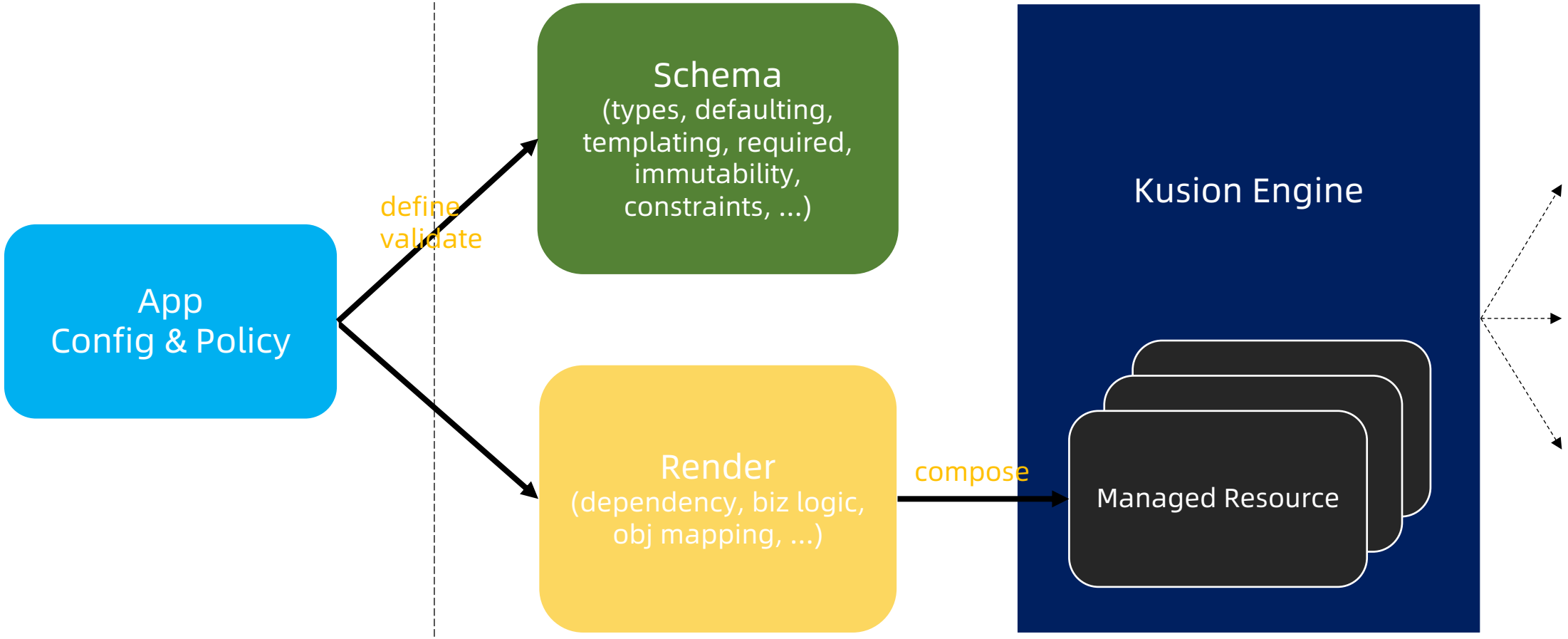
Enablers



Providers

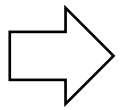
Platform

View of User Workspace

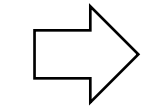


View of Automation

```
import base.pkg.kusion_models.kube.frontend

appConfiguration: frontend.Server {
    image = "howieyuen/gocity:latest"
}
```



```
schema ServerBackend[inputConfig: server.Server]:
    """ServerBackend converts the user-written front-end model `Server` into a
    collection of kubernetes resources and places the resource collection into
    the `kubernetes` attribute.
    """
    mixin [
        # Resource builder mixin
        mixins.NamespaceMixin,
        mixins.ConfigMapMixin,
        mixins.SecretMixin,
        mixins.ServiceMixin,
        mixins.IngressMixin,
        mixins.ServiceAccountMixin,

        # Monitor mixin
        pmixins.MonitorMixin
    ]

    # Store the input config parameter, ensure it can be seen in protocol and m
    config: server.Server = inputConfig
    # Workload name.
    workloadName: str = "{}{}".format(metadata.__META_APP_NAME, metadata.__META
    # App variable contains labels, selector and environments.
    app: utils.ApplicationBuilder = utils.ApplicationBuilder {}
    # Main containers and sidecar contrainers.
    mainContainer: {str:}
    sidecarContainers?: [{str:}]
    initContainers?: [{str:}]

    if config.mainContainer:
        assert config.image, "config.image must be specified and can't be empty
        # Construct input of converter using the volumes.
        mainContainer = utils.VolumePatch(config.volumes, [utils.ContainerFront
            **config.mainContainer
            if config.mainContainer.useBuiltInEnv:
                env += app.envs
            name = config.mainContainer.name or "main"
            image = config.image
            resource = config?.schedulingStrategy?.resource
        }))?[0]

    if config.sidecarContainers:
        sidecarContainers = utils.VolumePatch(config.volumes, [utils.ContainerF

    if config.initContainers:
        initContainers = utils.VolumePatch(config.volumes, [utils.ContainerFron

    # Construct workload attributes.
    workloadAttributes: {str:} = {
        metadata = utils.MetadataBuilder(config) | {
            name = workloadName
        }
        spec = {
            replicas = config.replicas
            if config.useBuiltInSelector:
                selector.matchLabels: app.selector | config.selector
            else:
```



Spec

Deployment

Service

ConfigMap

Database

Monitor

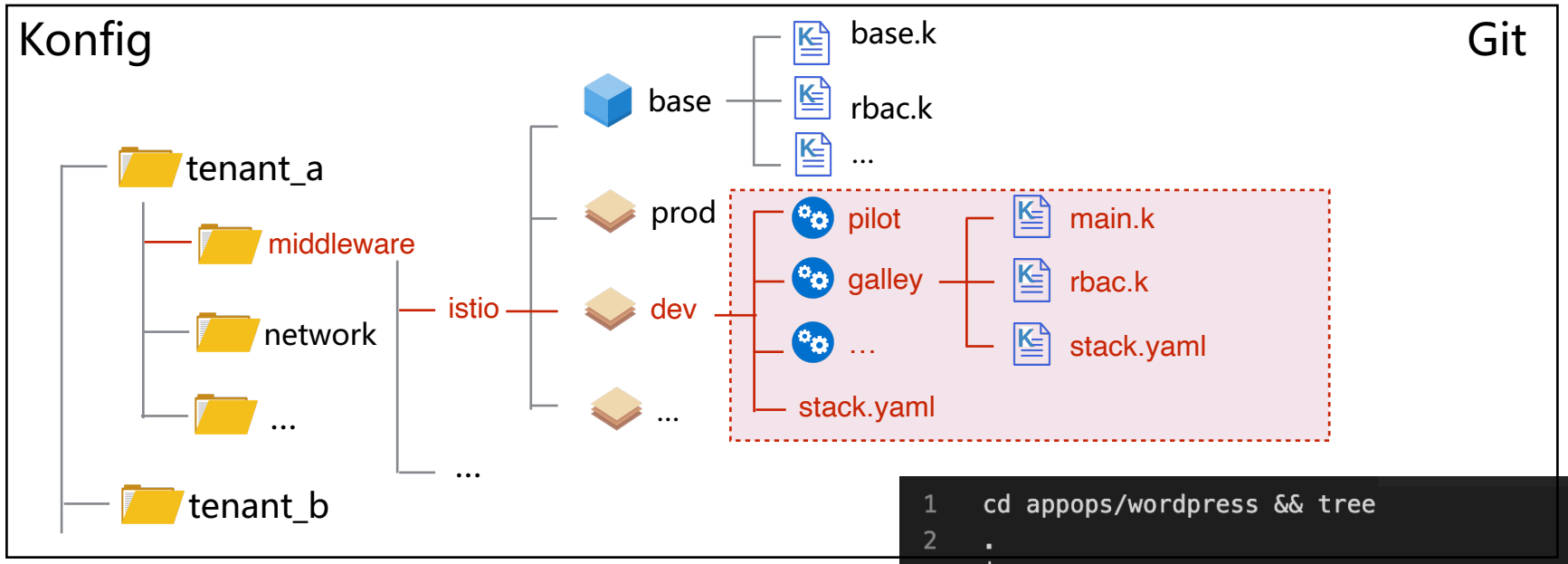
... ...

App Team Space

Platform Team Space

Multi-tenant, Multi- scenario, Multi-cloud

Centrally defined, globally delivery




```
cd appops/wordpress && tree
.
├── README.md
├── base                            // Common configuration for all stacks
│   └── base.k
├── dev                             // Stack directory
│   ├── ci-test                     // Test data
│   │   ├── settings.yaml
│   │   └── stdout.golden.yaml
│   ├── kcl.yaml                    // Compile configuration for current stack
│   ├── main.k                      // App Configs matained by App Dev
│   ├── platform.k                  // App Configs matained by Platform Dev
│   └── stack.yaml                  // Stack metadata
└── project.yaml                    // Project metadata
```

Build it, Run it

With to-dev product

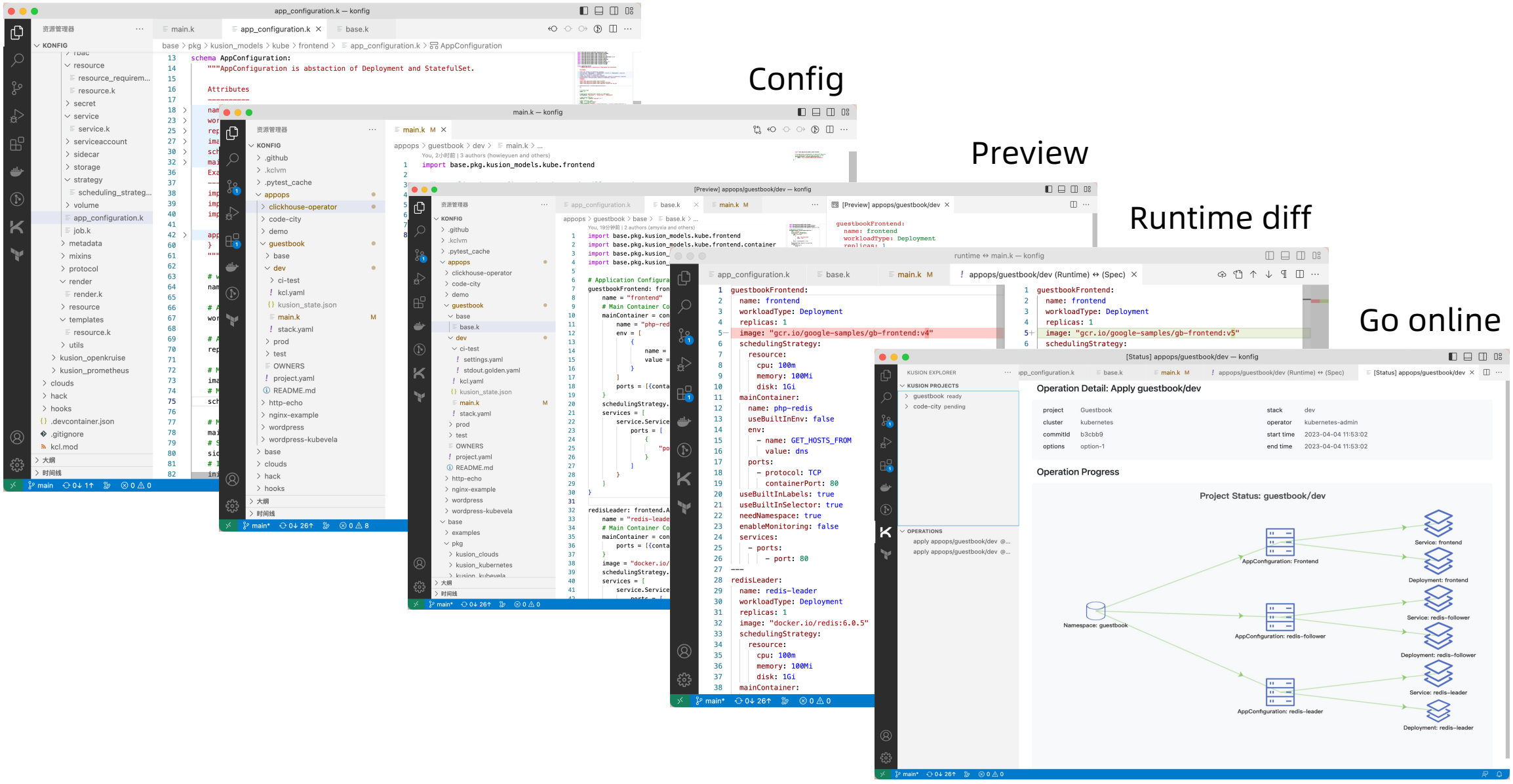
Abstract

Config

Preview

Runtime diff

Go online



Practice in AntGroup

Practice

Efficiently enable business success

ANT GROUP

1K/day	10K+/day	1 : 9	100K+
Pipelines	KCL Compilations	Plat : App Dev	Commits
600+	2500+	1M+	10M+
Contributors	Projects	KCL Codes	YAML

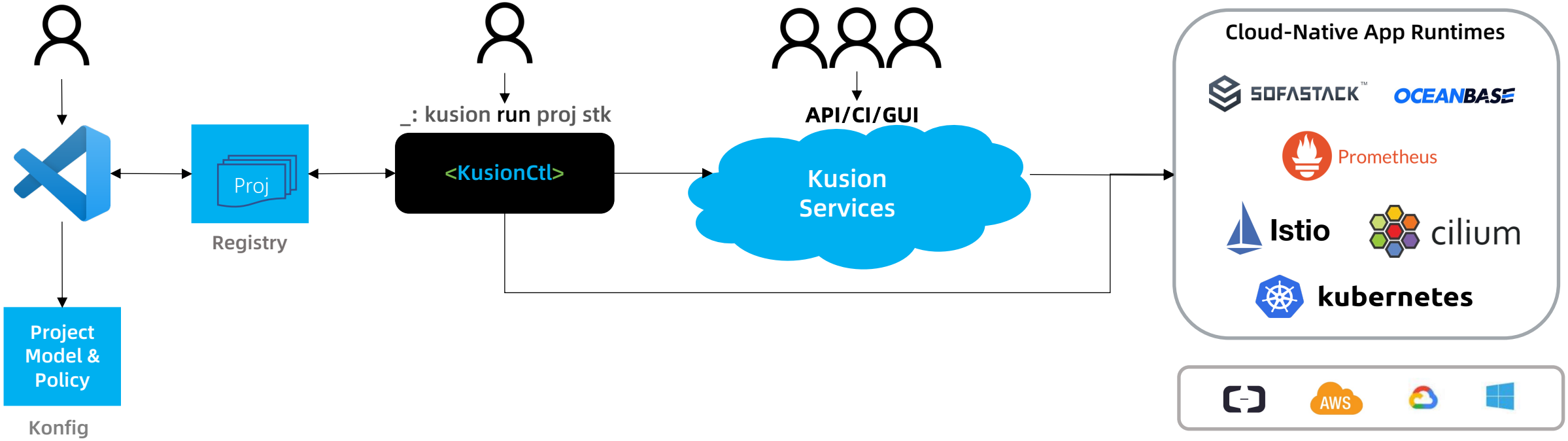
HUAWEI

KYLIGENCE®

有赞

Future

Next Stage



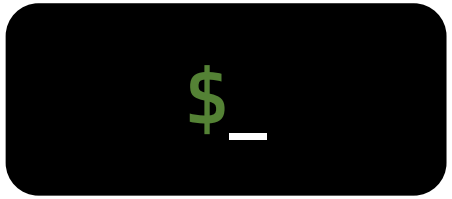
- Project-based
- Role-based Authority
- Write, Commit, Publish
- Indexing
- Versioning
- Hosting
- Identity-based 2A
- Credential Mgmt
- Hierarchy Control
- Install & Preview
- State Mgmt
- Orchestration
- Hybird-Resource
- Provision & Watch
- History & Audit
- Multi-{Cloud, Cluster}
- Tracing
- Troubleshooting
- Health & Event Aware

Organize | Code | Run

Unleash Platform Potential

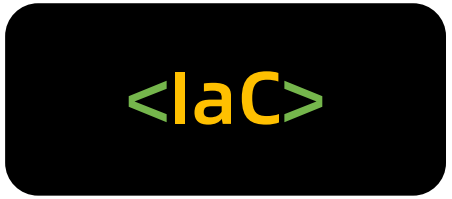
KUSION STACK

Scripting



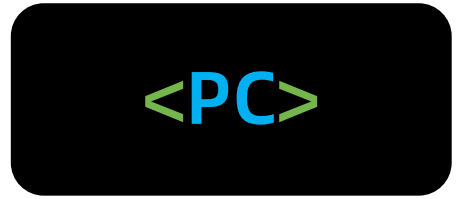
Infra as Code

Clouds



Platform Collaboration

K8s, Services, Clouds



- Imperative Commands

- Resource Centric, for Ops
- Managed State and Provision
- Cloud API Native

- **App Centric, for Devs**
- **Unified Organization and Operation**
- **Abstraction, Validation, Scalability**
- **Kubernetes Control Plane Native**
- **Hybrid Resource Automation**
- **Self-Service**

Tech Roadmap

KCL

v0.4.6

- Ease of Use and Error Improvement
- KPT/Kustomize/Helm KCL Plugin
- Lighter Go SDK
- IDE Extension Pre-release
- Playground

v0.5.0

- Grammar and Standard Library Improvement
- Kubectl KCL Plugin
- KPM Release
- KCL IDE Extension Release
- More SDKs e.g., Java

v0.6.0

- New KCL UI & Tool Design
- Helmfile KCL Plugin
- KRM KPM Integration
- More KPM Registry Integrations
- More IDE Extensions e.g., Vim
- CompilerBase Core Components

v0.7.0

- New KCL UI & Tool Release
- CompilerBase Grammar and Sematic Components
- Module Marketplace
- IDE Product

Language
Tools & IDE
SDK
Integration
Subproject

2023.3

2023.6

2023.9

2023.12

v0.7

- **Kusion (Resource):** Hybrid resource operation like Terraform and Kubernetes in an unified way
- **Kusion (Resource):** Kubernetes native resource health check
- **Quality :** Kusion E2E test framework

v0.8

- **Konfig (Model):** Support Aliyun RDS, AWS RDS
- **Konfig (Toolbox):** Structure validation
- **Kusion (Resource):** Customimze resource health check
- **Security :** Secret as Code
- **IDE:** Kusion Operations Integration

v0.9

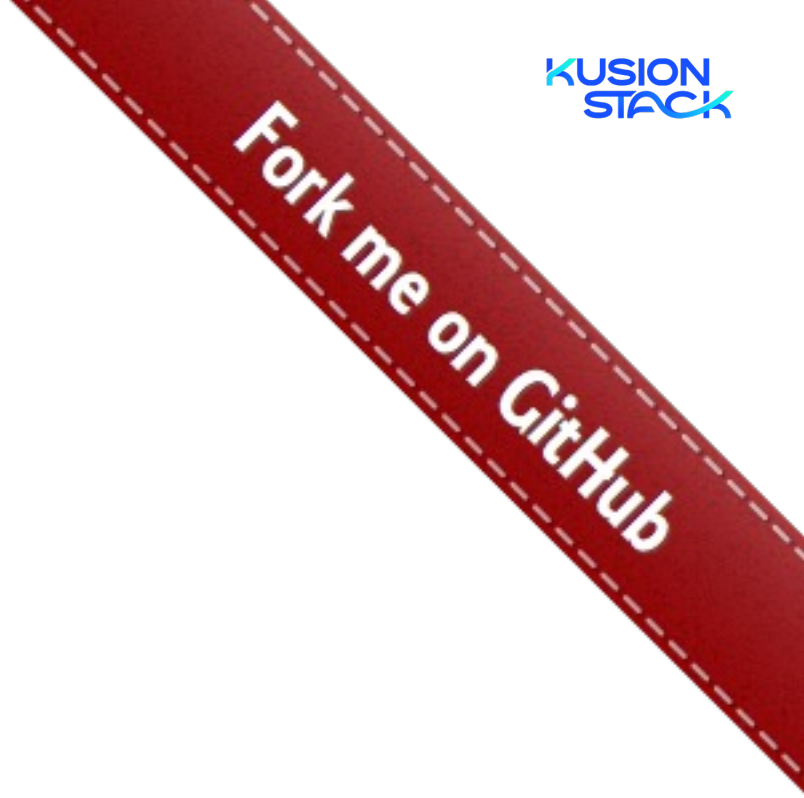
- **Konfig (Model):** Support Azure SQL Database
- **Konfig (Toolbox):** Dependency analysis
- **Kusion (Operation):** App Progressive Rollout
- **Kusion (Config Framework):** K8s raw YAML

v0.10

- **Konfig (Model):** Support Aliyun SLB, AWS ELB
- **Kusion (Operation):** Custom Pipelines
- **Kusion (Operation):** Flexible Kusion Runtime
- **Kusion (Operation):** Operation REST
- **Kusion (Config Framework):** Helm

Kusion & Konfig

Resources



- Web Site
  - https://kusionstack.io/

- Source Code
  - https://github.com/KusionStack/kusion
  - https://github.com/KusionStack/kcl
  - https://github.com/KusionStack/konfig

- Contact
  - https://github.com/KusionStack/community#contact
  - https://github.com/KusionStack/community

- Twitter
  - @KusionStack

Thank you

KusionStack Team