KusionStack Origin, present and future

KusionStack Team

Agenda

Origin

Cloud-native technologies

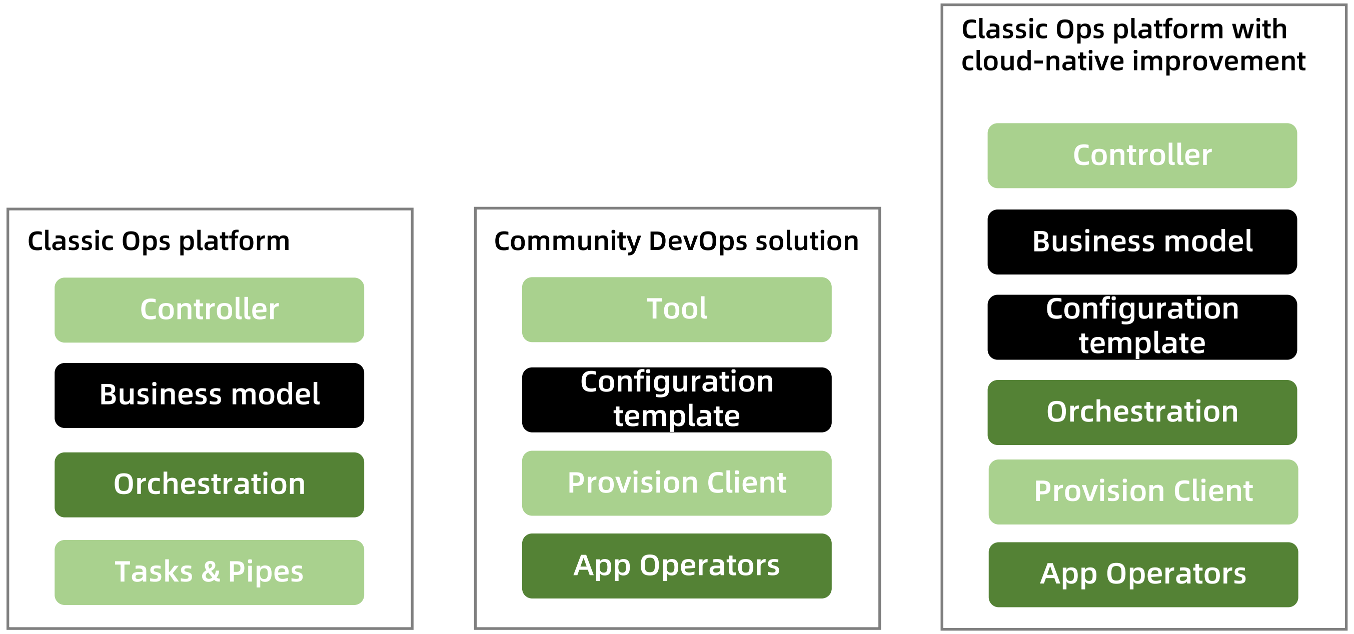
Are eating the world

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

Diversity, scale and change

Create ongoing challenges

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



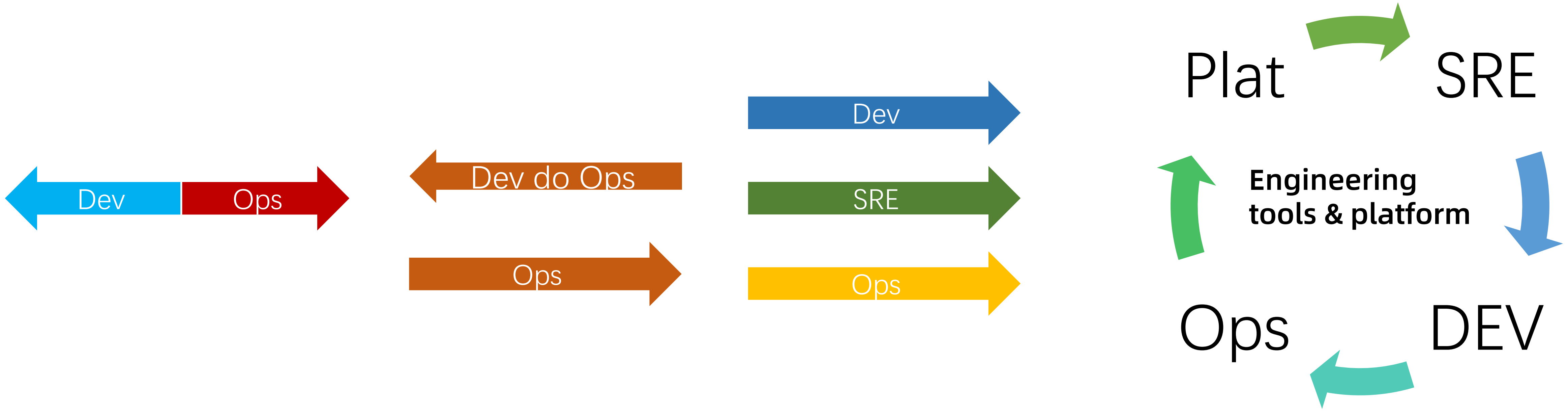
What we tried and not that successful

The darker the part, the more complex, the faster the change

Goal

Effective Teamwork

Enable overall success

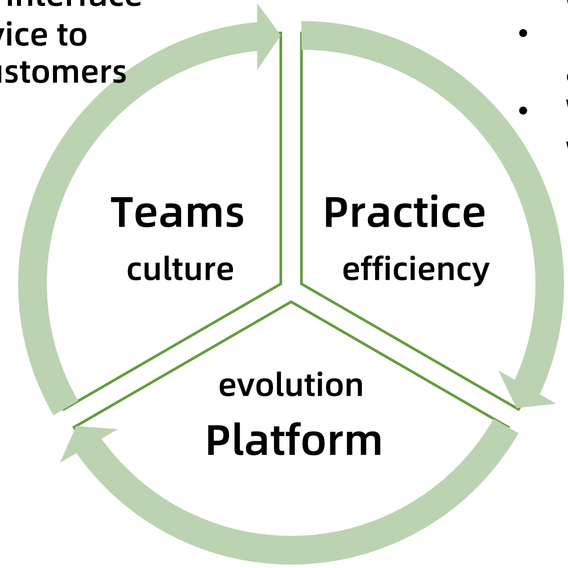


Collaborate, Automate

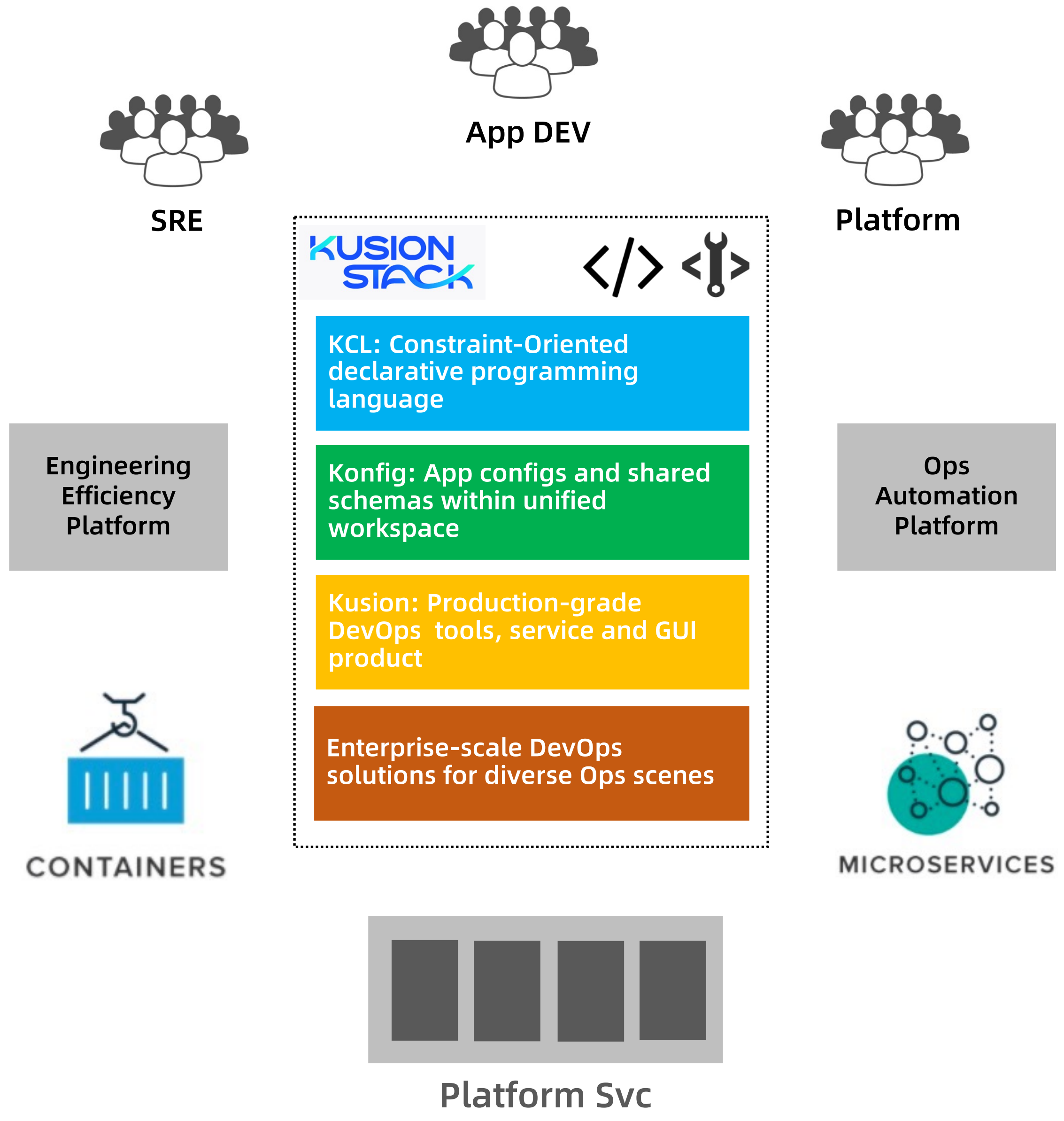
Make scaled DevOps possible

- 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


SRE
App DEV
Platform
KUSION STACK
KCL: Constraint-Oriented declarative programming language
Konfig: App configs and shared schemas within unified workspace
Kusion: Production-grade DevOps tools, service and GUI product
Enterprise-scale DevOps solutions for diverse Ops scenes
Engineering Efficiency Platform
Ops Automation Platform
CONTAINERS
MICROSERVICES
Platform Svc

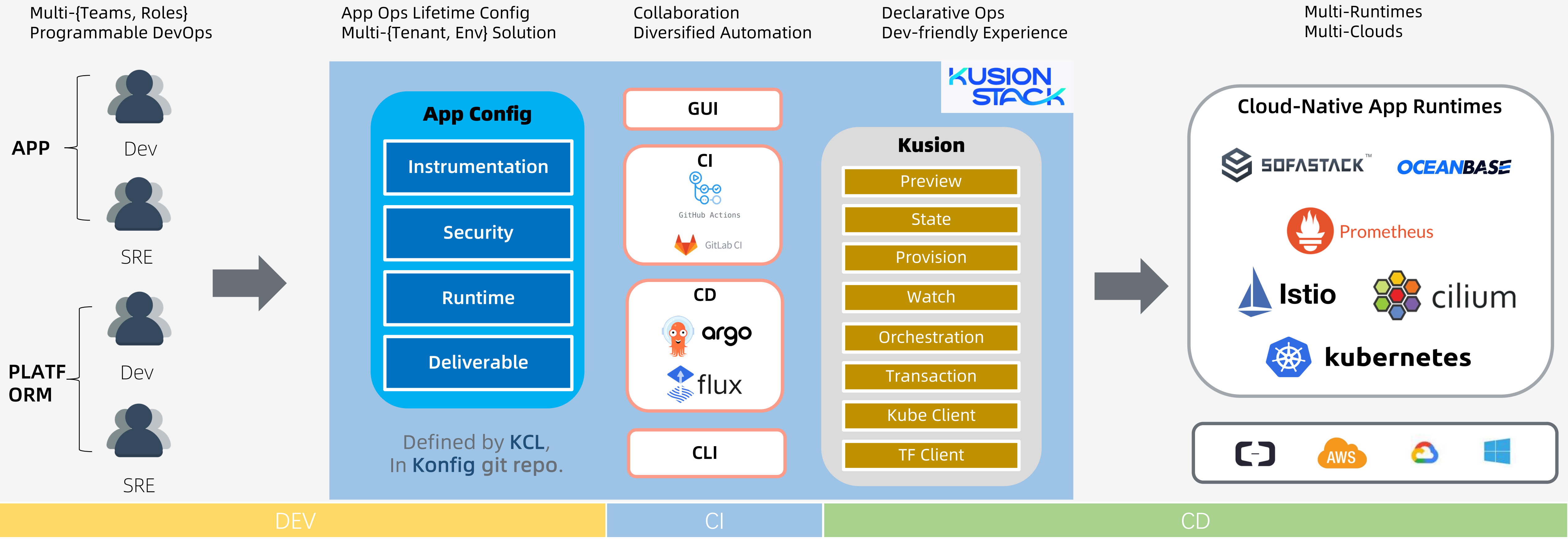
A Stack to Deliver Value

Make scaled delivery agile

Enterprise Declarative DevOps

App Centric Shipping Anywhere

Codify Stack for Platform Engineering

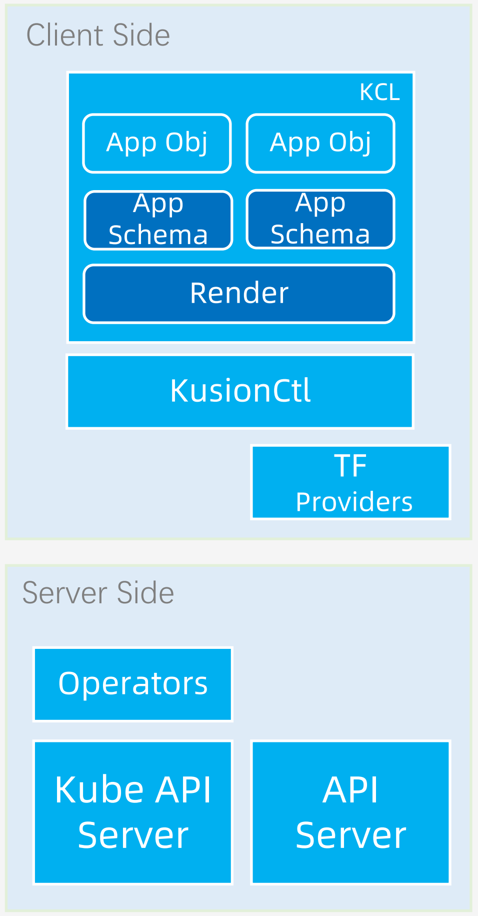


Solution

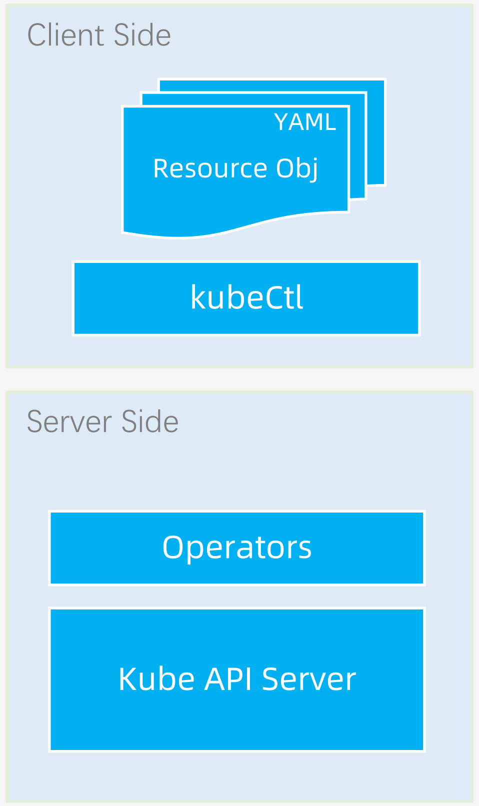
Kube Ops with X

Portable client solution with app centric interface

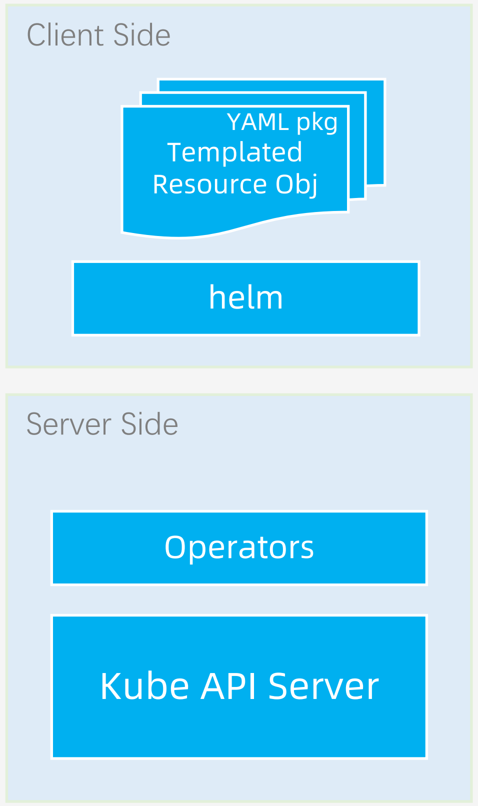
- **App Centric**
  - Modeling
  - Abstraction
  - Customization
  - Combination
  - Policy
- **Pure Client Solution**
  - Codify
  - Lightweight
  - Flexible
  - Scalable
  - Portable
  - Left-shifted stability
- **Hybrid-platform**
  - On Kube & TF
  - On Multi-Clouds
  - Provision
  - Orchestration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



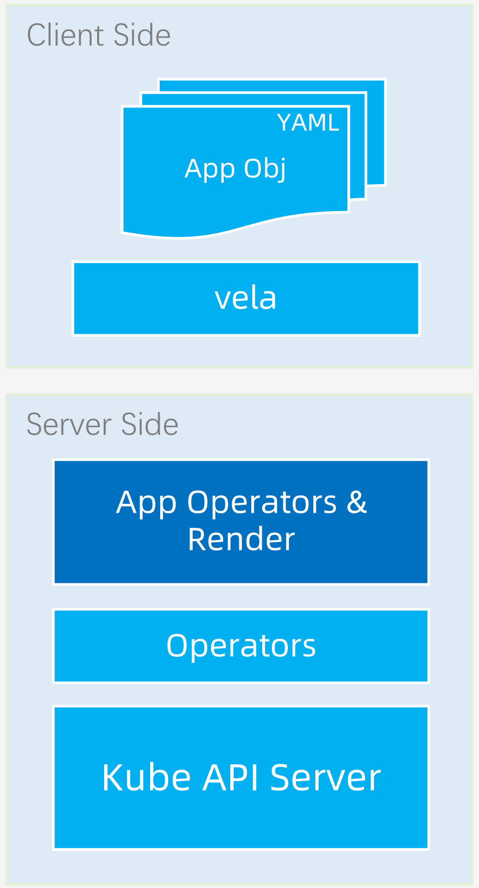
➢ **KusionStack**



➢ **Typical tool：Kustomize**



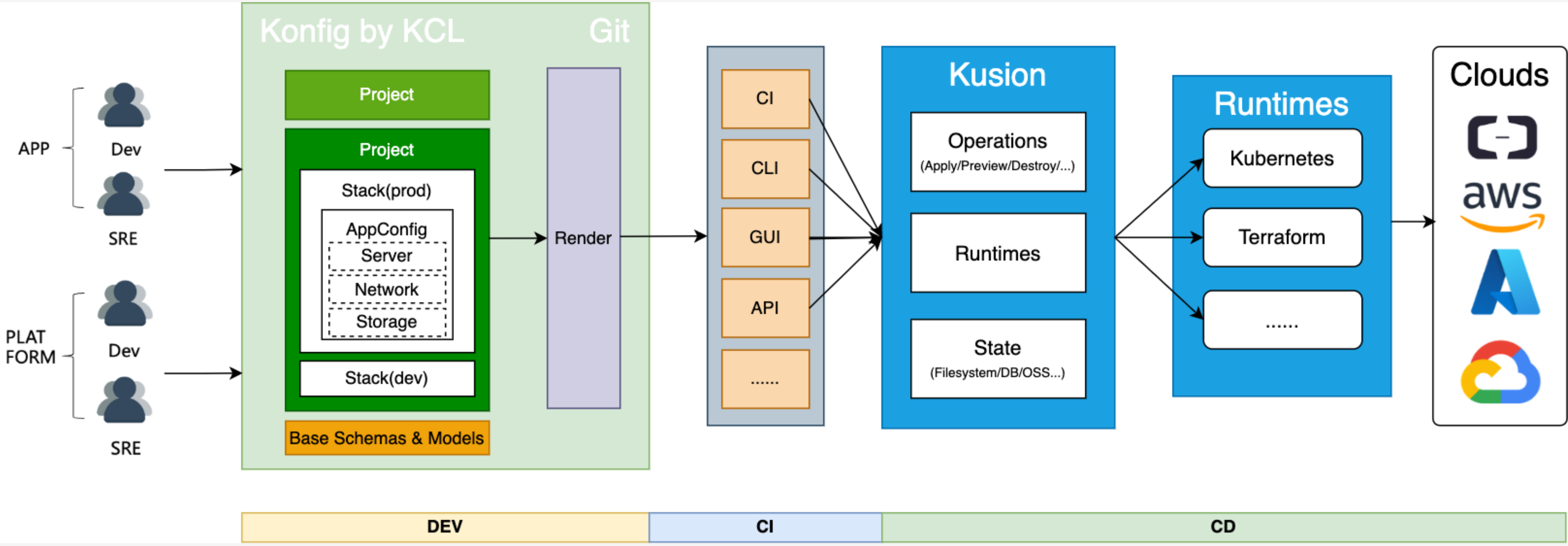
➢ **Typical tool：Helm**



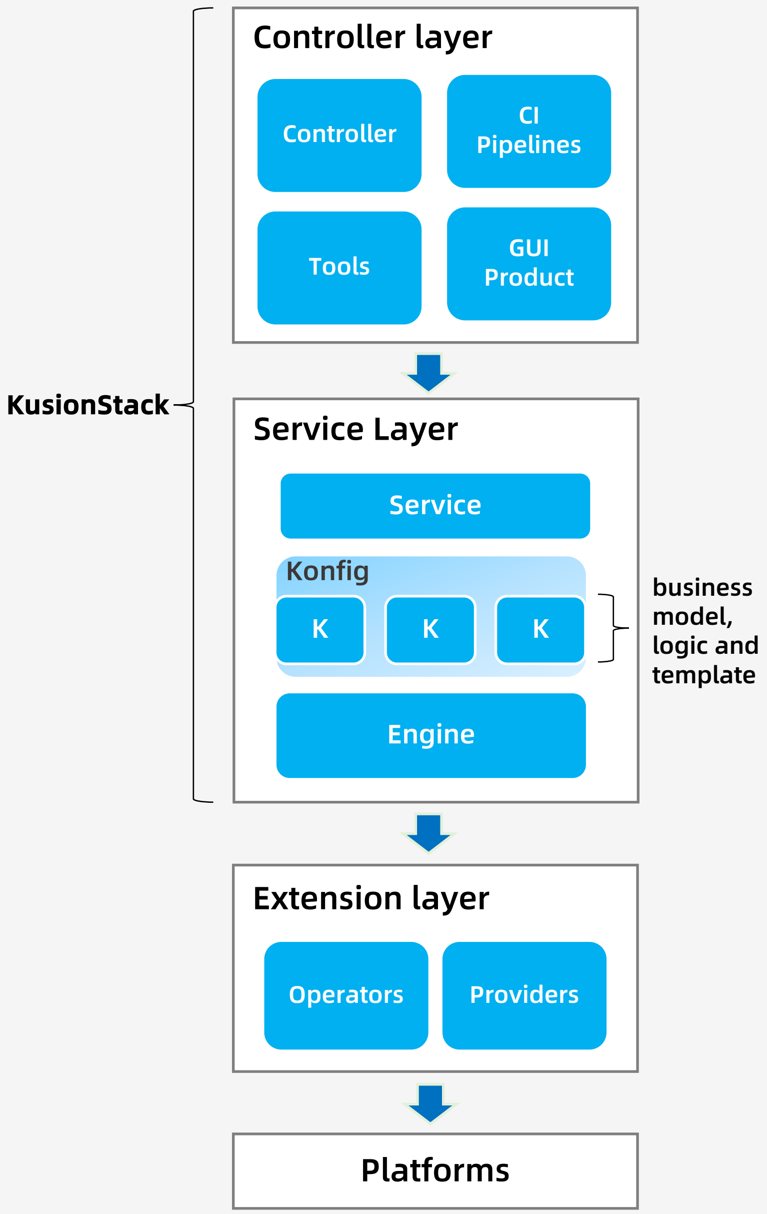
➢ **Typical tool：KubeVela**

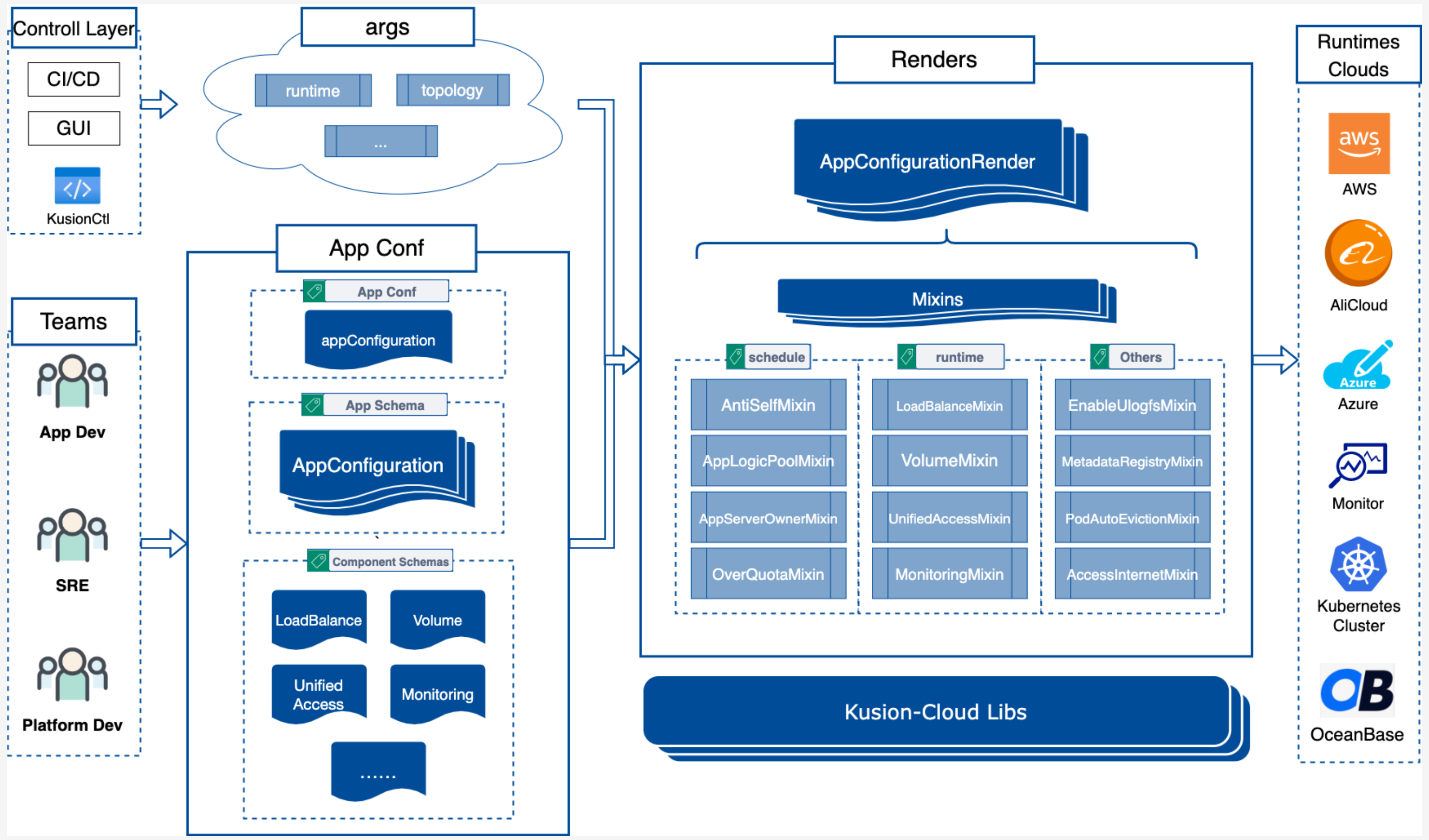
Enterprise Solutions

Scaled and flexible development and automation

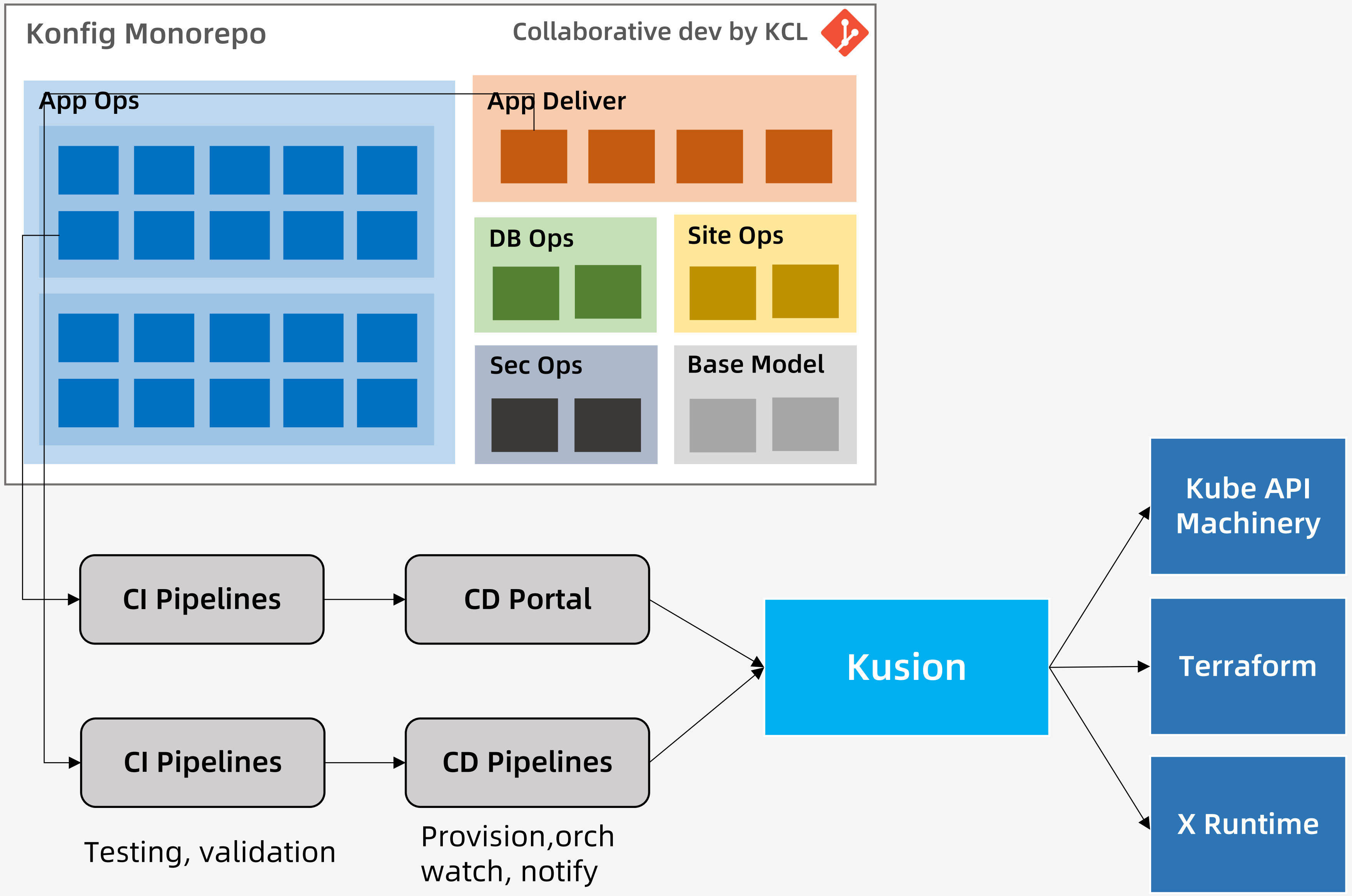


Layers & Collaboration





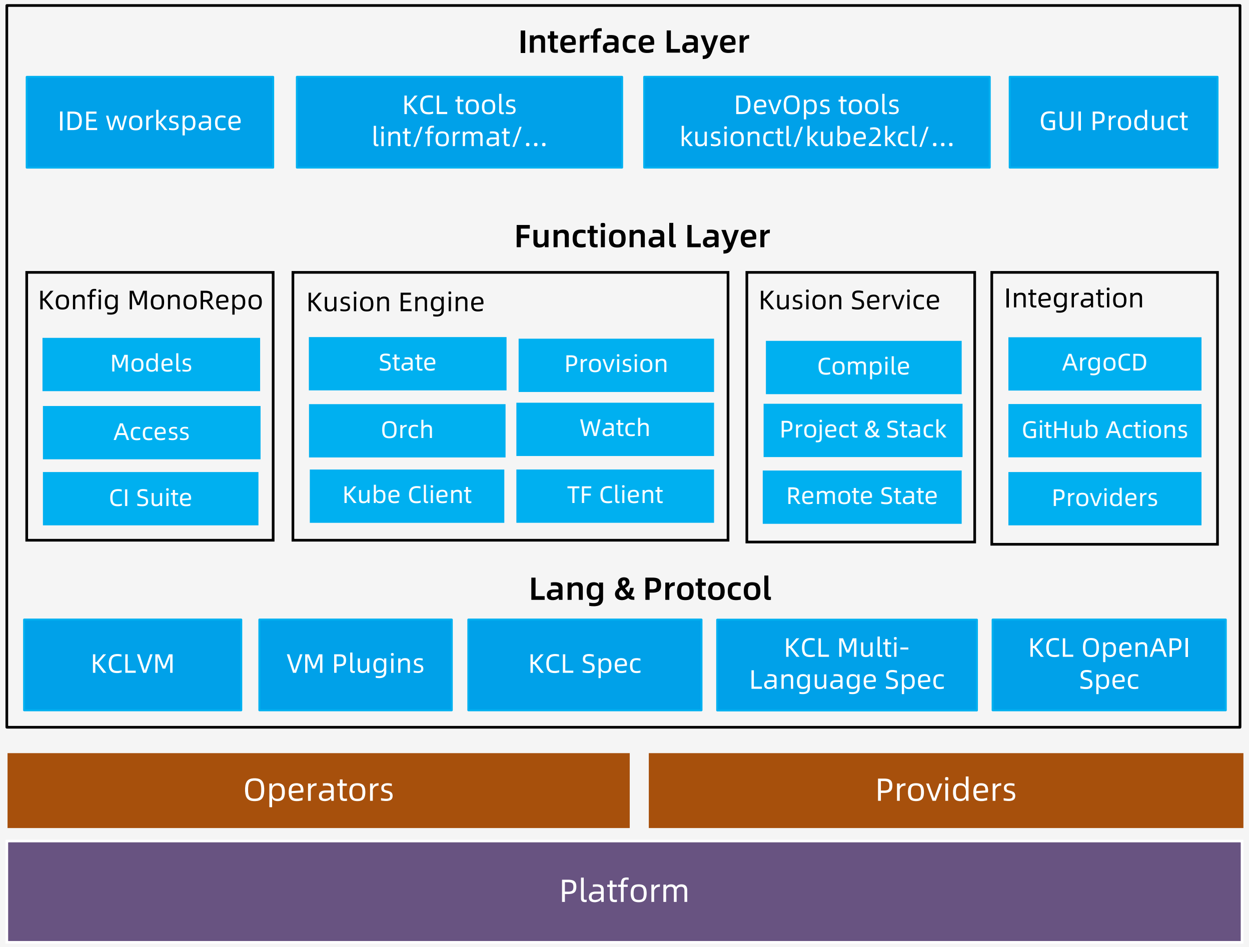
Automation Workflow

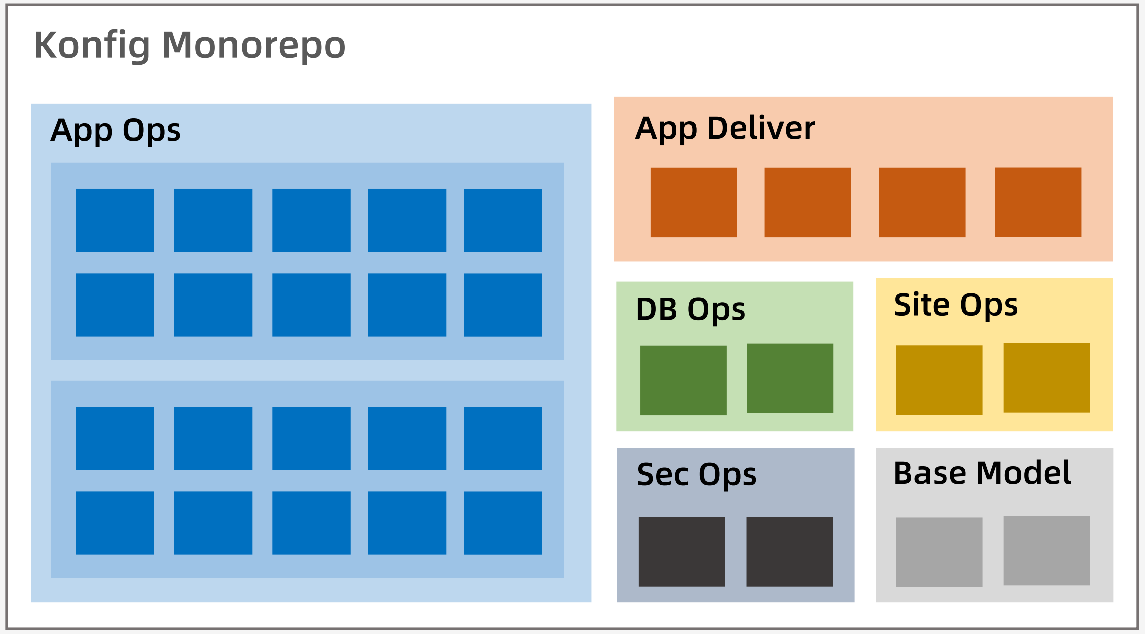


Tech

KusionStack Arch

Lang, tools, interface and workspace





KusionStack - 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

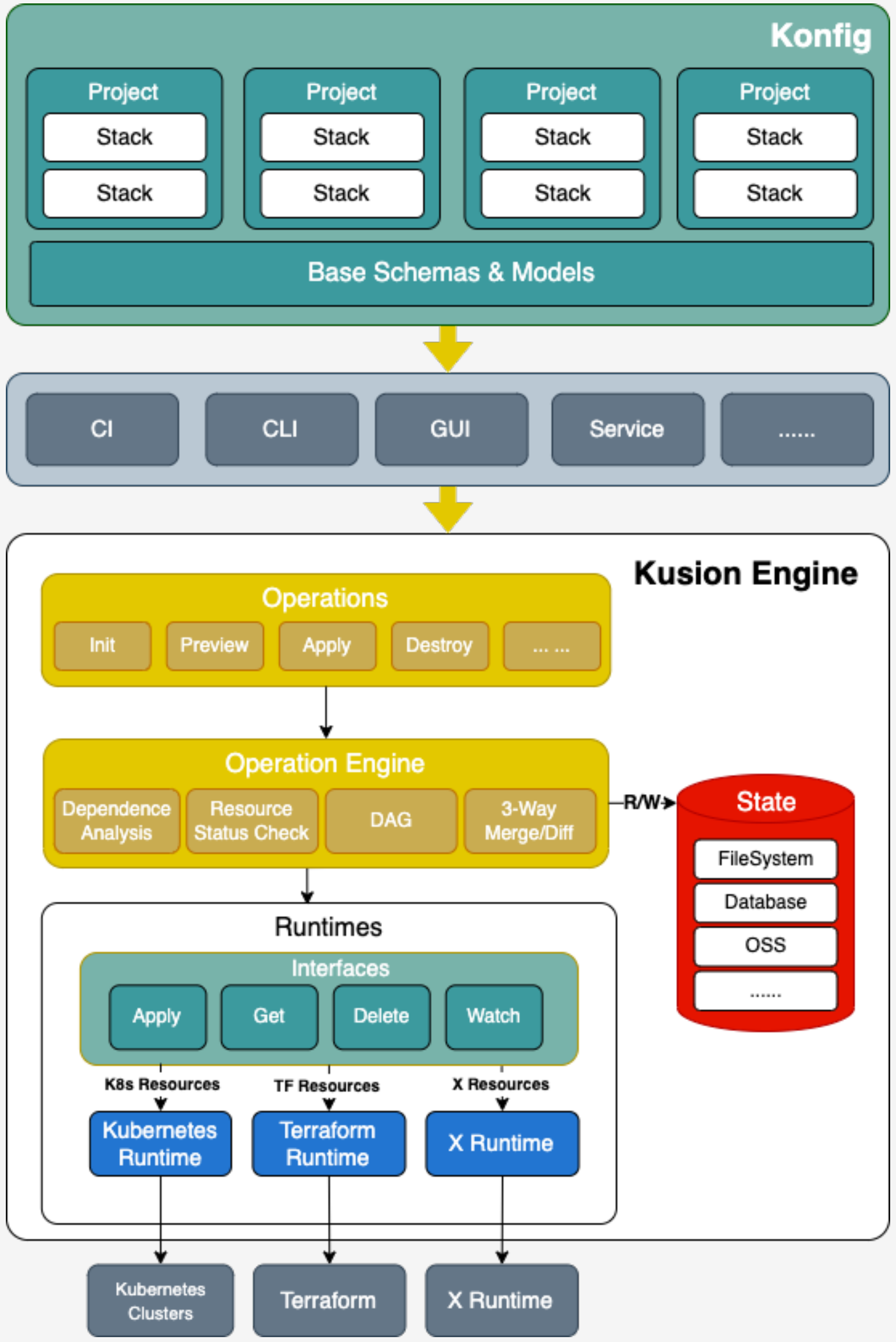
Konfig & Kusion

Managed resource across multiple runtimes

Operation Engine: provide core features to support all Kusion operations

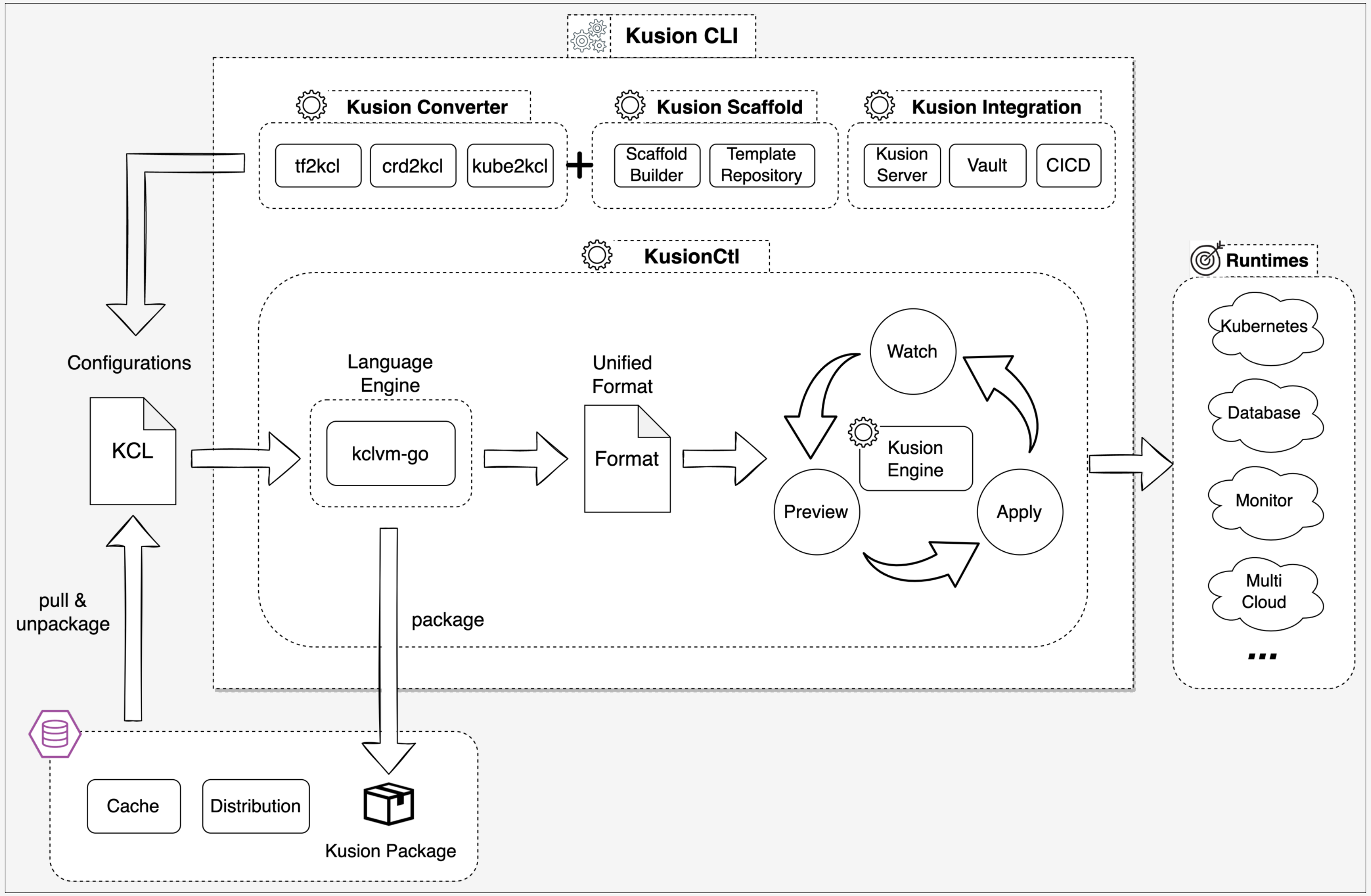
Runtimes: represent actual infrastructure runtimes managed by Kusion.

State: a mapping between resources in Konfig and the actual infra resource



Kusion Tools

Delivery workflow easier



KusionStack - KCL

KCL - An Open Source Constraint-Based Record & Functional Language

Well-Designed

Spec-driven
Config, Schema,
Lambda, Rule

Easy to Use

In Configuration
Policy cases

Modeling

Schema-Centric
Abstraction

Stability

Static Type System
Constraints
Rules

Scalability

Separated Config Blocks
Rich Merge & Override
Strategies

Automation

CRUD APIs
Multi-Lang SDKs
Plug-ins

Cross-Platform

High-Performance
Multi-Runtime

Cloud-Native Affinity

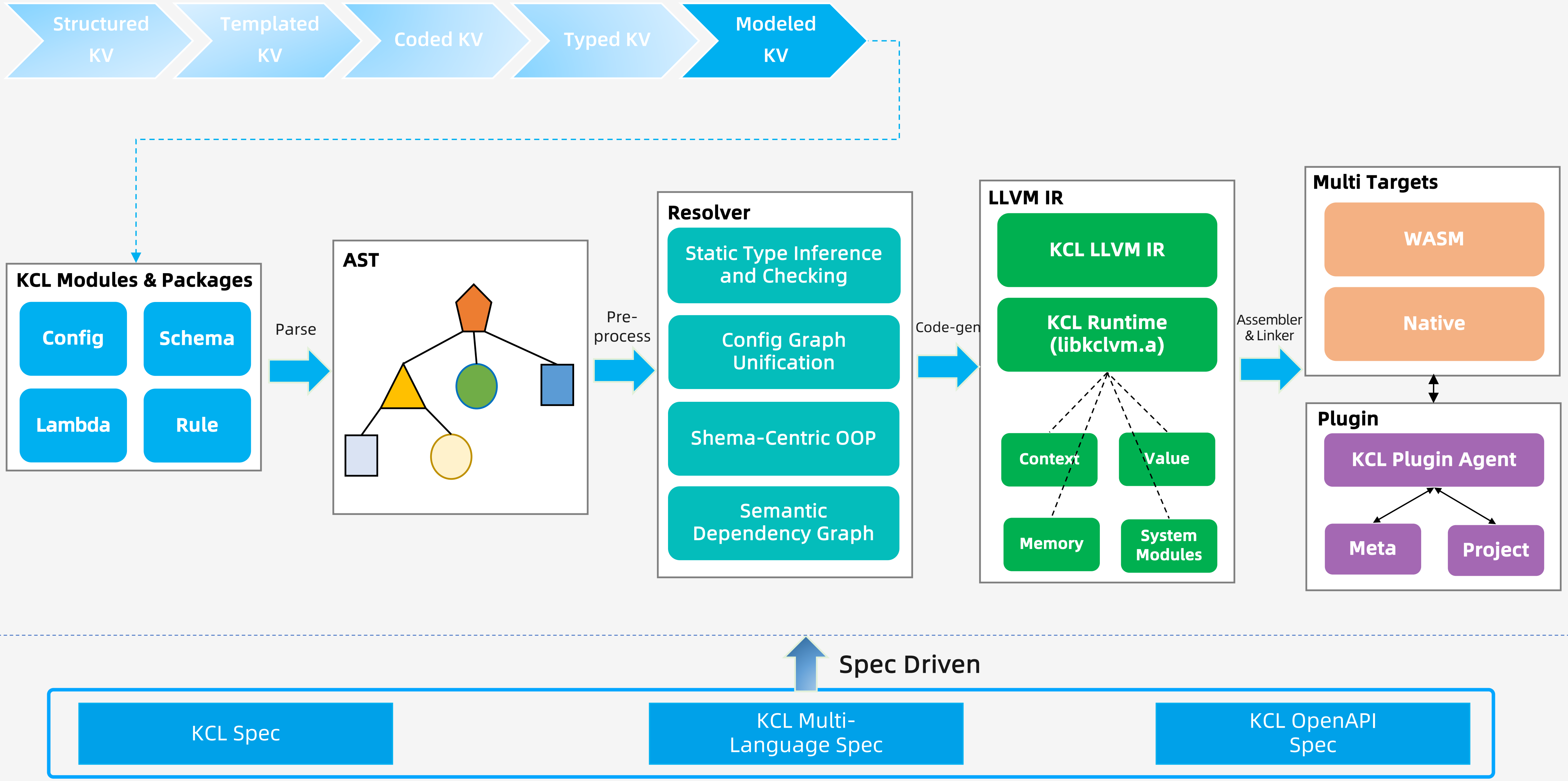
Open API/CRD
Specs/YAML Spec

Dev Friendly

Lint/Test/Vet/Doc Tools
VS Code/Intellij IDE

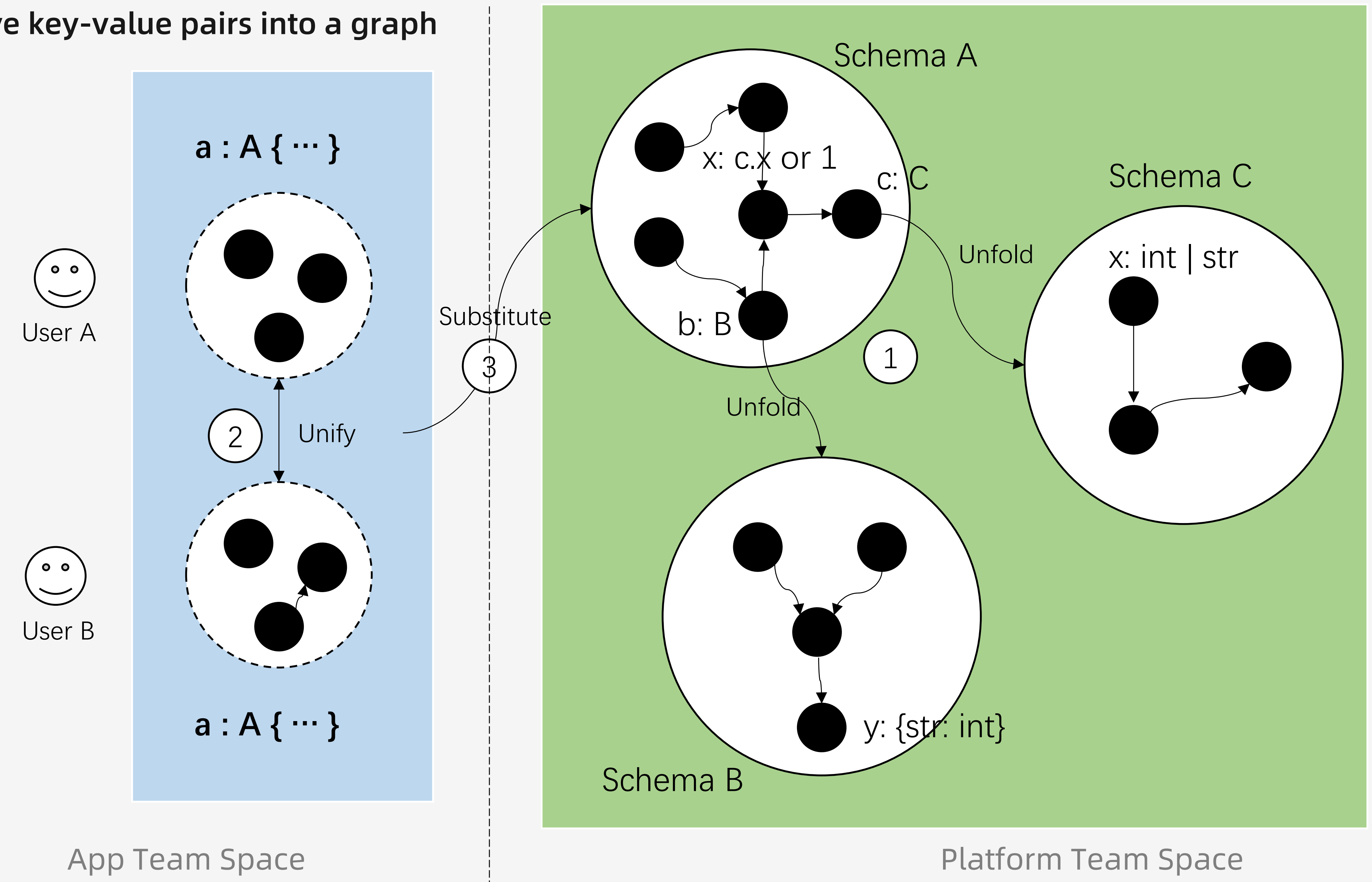
KCL

Config, Schema, Lambda, Rule



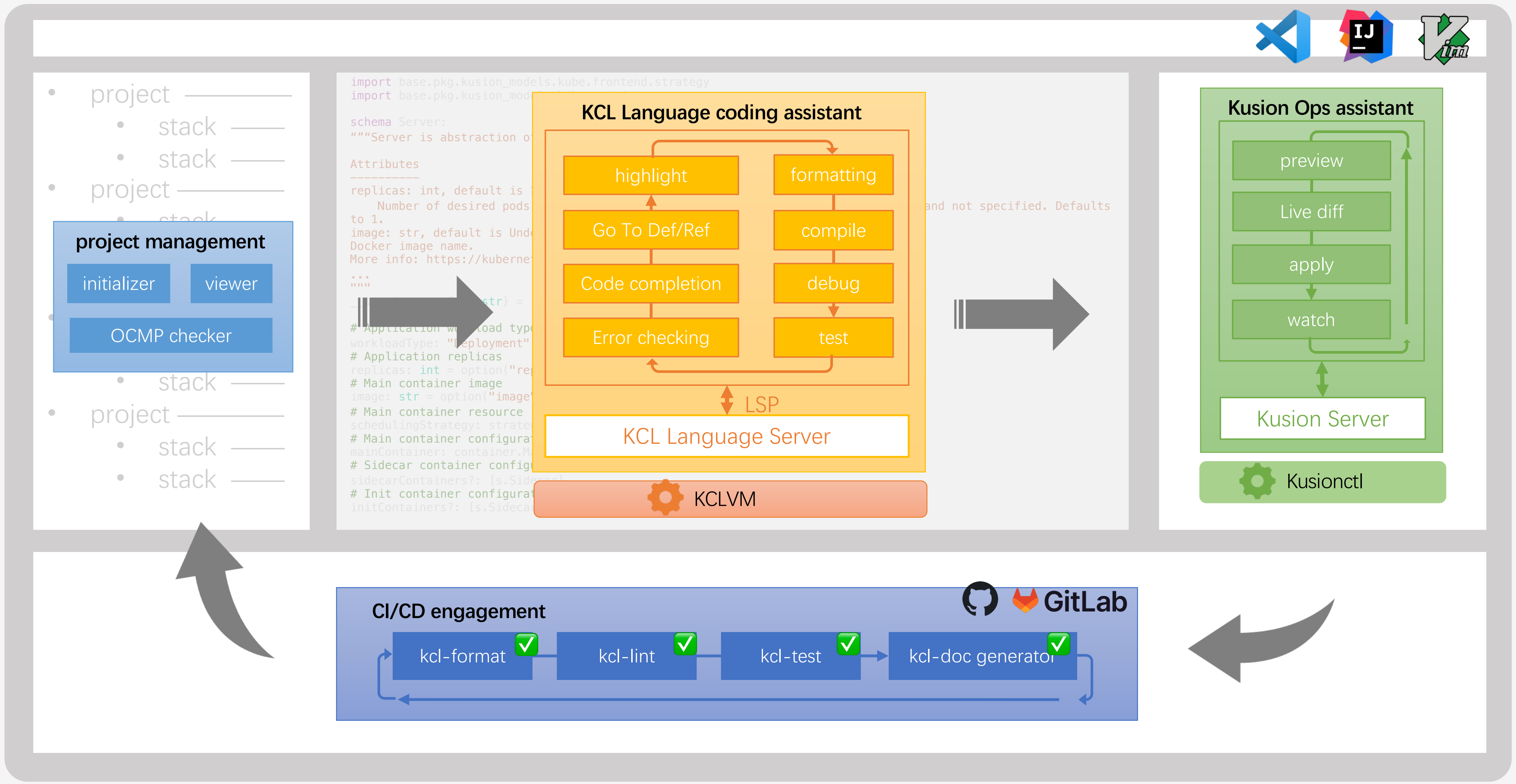
KCL Internal Graph Model

Weave key-value pairs into a graph



KCL Tools & IDE Workspace

Make Ops collaborative coding and work happy



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

schema Server:
"""Server is abstraction o

Attributes
----------
replicas: int, default is
    Number of desired pods                                    and not specified. Defaults
to 1.
image: str, default is Und
Docker image name.
More info: https://kuberne
...
"""

# Application workload typ
workloadType: "Deployment"
# Application replicas
replicas: int = option("re
# Main container image
image: str = option("image"
# Main container resource
schedulingStrategy: strate
# Main container configura
mainContainer: container.M
# Sidecar container config
sidecarContainers?: [s.Sidecar]
# Init container configurat
initContainers?: [s.Sideca
```

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

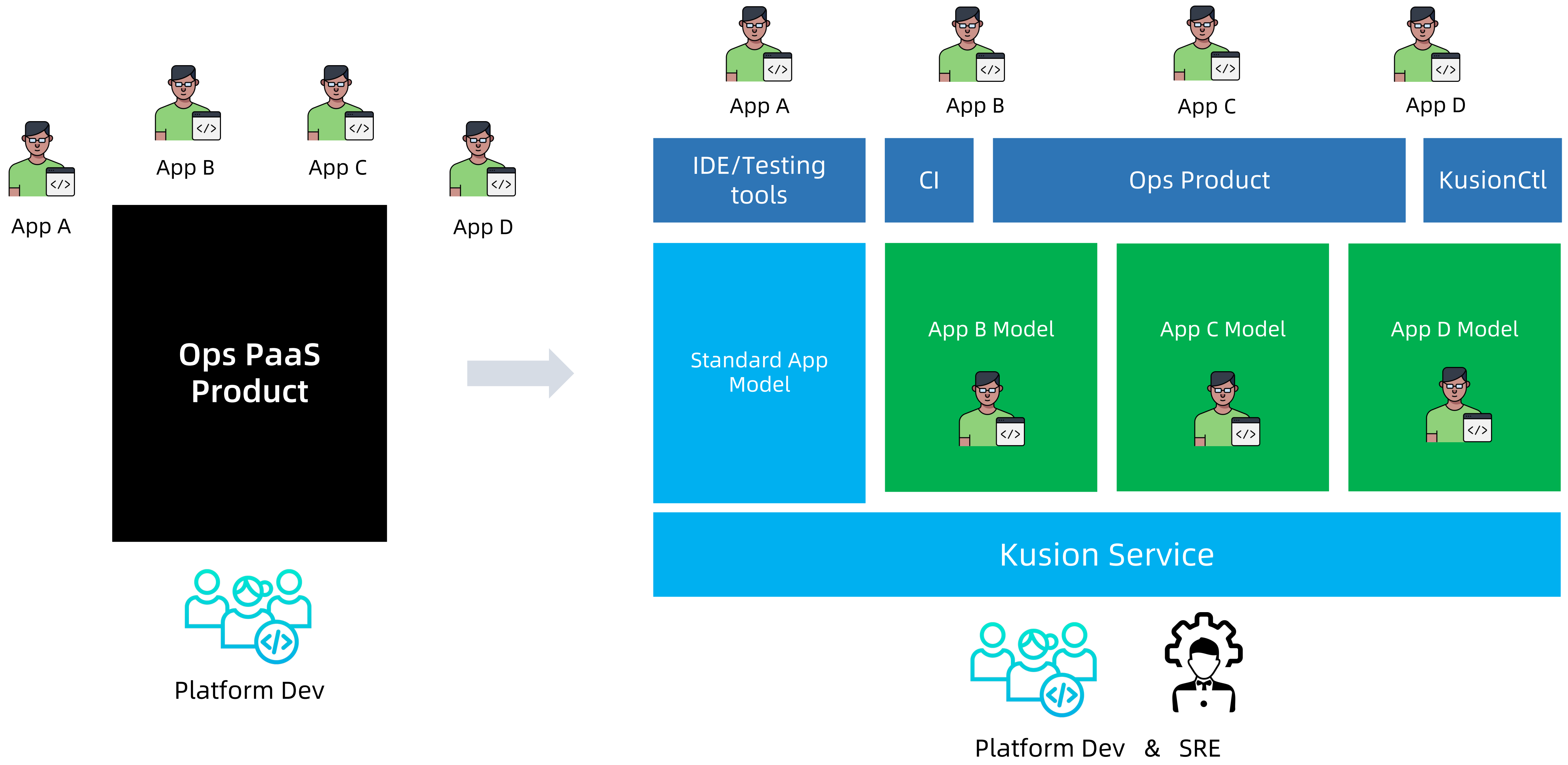
Goals
- Deliver platform projects to multi-clouds
- Enable user-side self-service and reduce ops and service costs

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

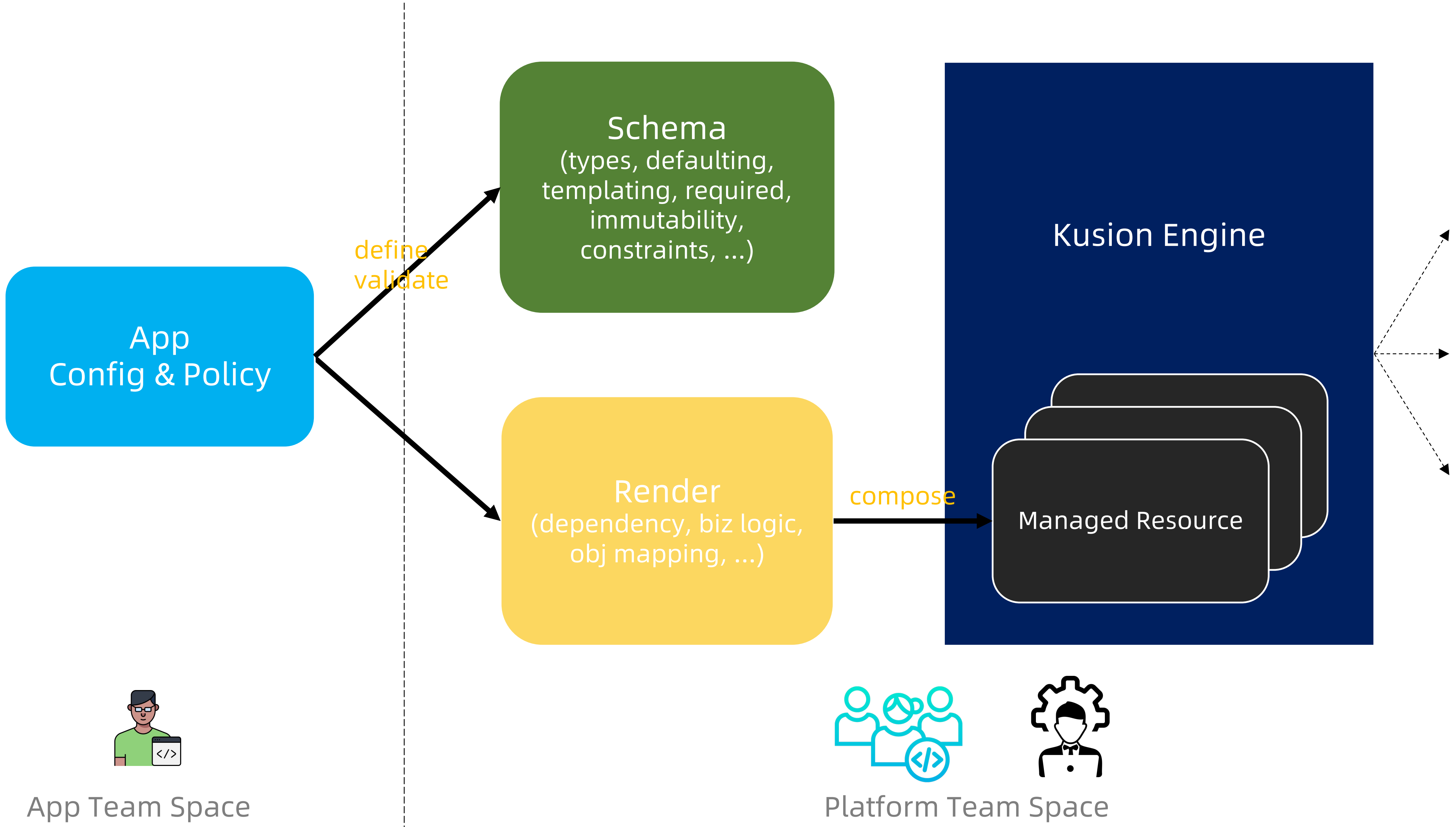
Pain points
- Unable to invest more time in R&D due to user supporting
- Unable to make app developers to access platform capabilities in a uniform, stable and low-cost way

Highly open self-service

Best fit application models & tools for various user cases

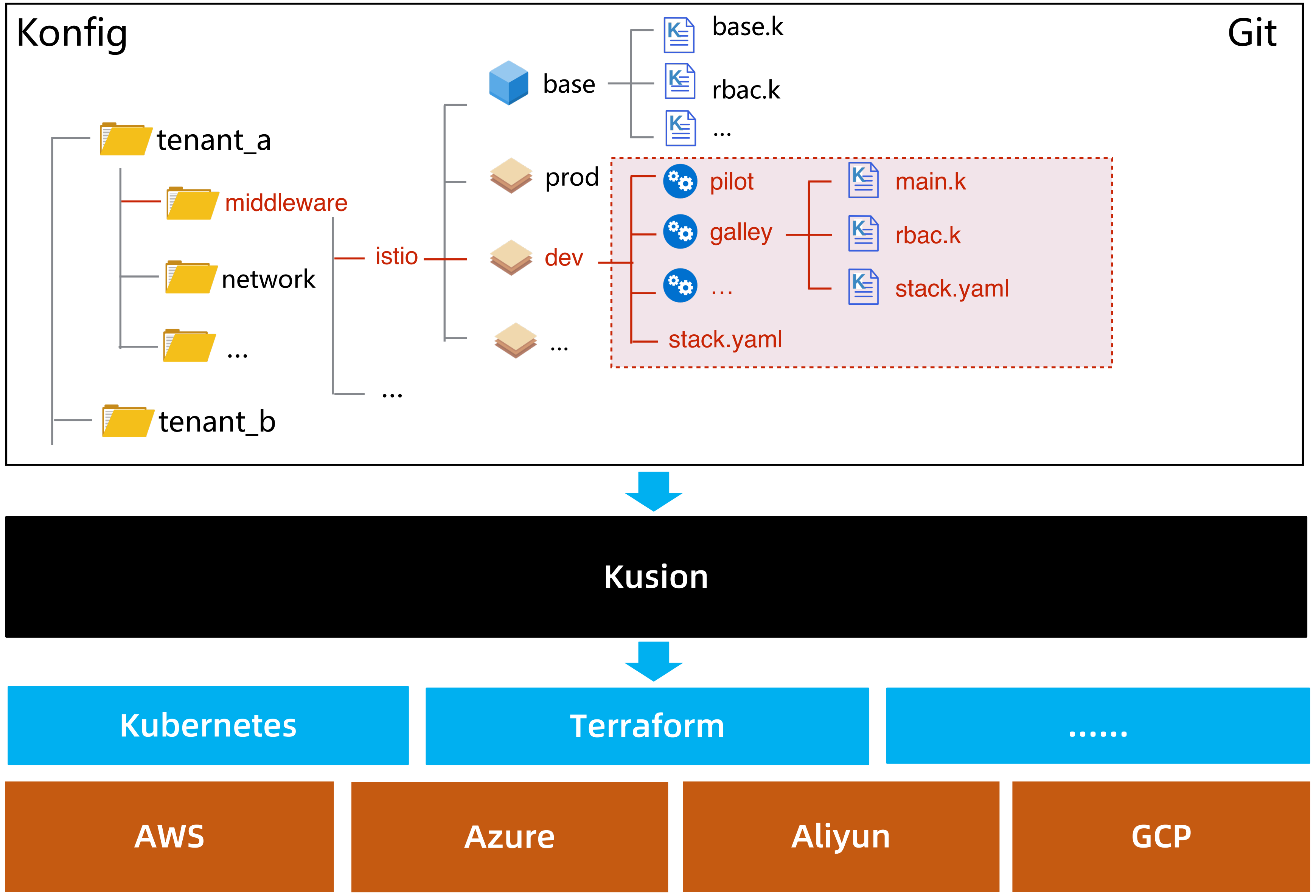


View of User Workspace


App
Config & Policy
define
validate
Schema
(types, defaulting, templating, required, immutability, constraints, ...)
Render
(dependency, biz logic, obj mapping, ...)
compose
Kusion Engine
Managed Resource
App Team Space
Platform Team Space

Multi-tenant, Multi- scenario, Multi-cloud

Centrally defined, globally delivery



Practice in AntGroup

Scaled

Ongoing app & infra delivery and ops

1500+

Projects

100+

Clusters

Practice

Efficiently enable business success

1K/day

Pipelines

10 K/day

KCL Compilations

1: 10

Plat: App Dev

6 scenarios

One-Stop

2 hours - 2 day

Feature Dev Period

300 - 400/day

Commits

Dev & SRE

Multi-Role DevOps

Hybrid cloud

Delivery

Culture

Precipitate engineering culture, share domain knowledge

~500	80000+	~22000	~800000
Contributors	Commits	PRs	KCL Code

Future

Unlocking Platform Value

Scripting

$_

1990's

- Imperative Commands

Infra as Code

(Terraform on Clouds)

<IaC>

2000's

- Programmable Key-values
- Limited Scalability
- Managed State and Provision

Platform Collaboration

K8s, Clouds + ?

<PaC>

2020's

- For Developers (App & Plat)
- Abstraction, Validation, Scalability
- Kubernetes Control Plane Native
- Hybrid Resource Automation
- Self-Service

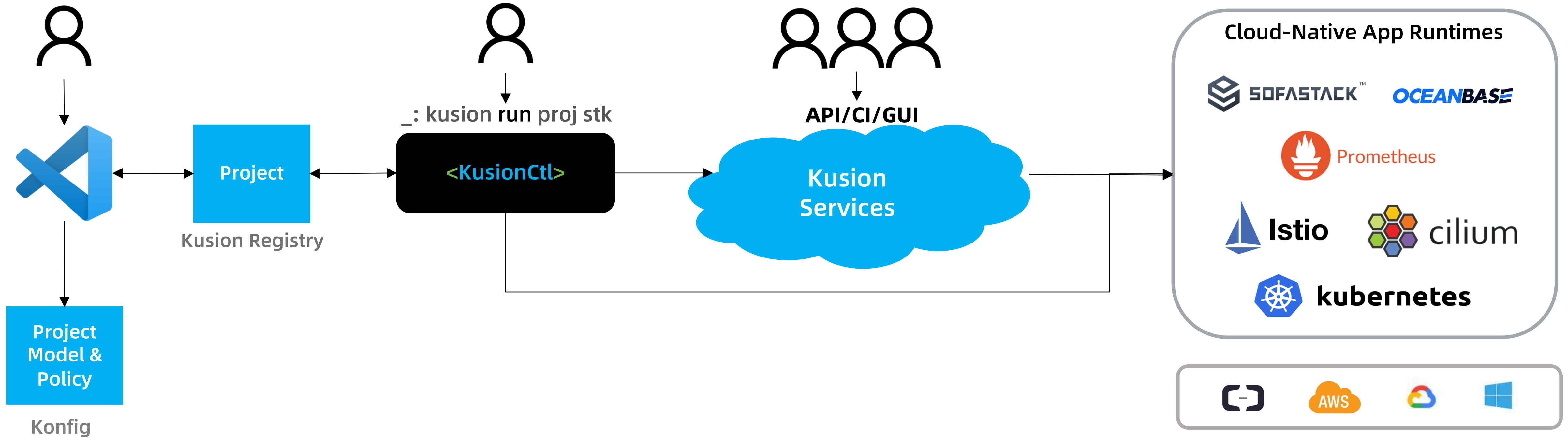
Next Stage

to dev
to team, to org

to share
to open source

fine grained
authorization

easy
self service



Org nize

- 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

Codify | Run

Tech Roadmap

KCL
- More Friendly for Dev
- Wider Ecological Integration
- Powerful Lang & Compiler Capabilities
- Advanced Technology Exploration

2022.9

v0.4.3
- Lang Simplification Stage 1
- KCL APIs by Rust
- Completely KCL Tools Support: lint, test, ...
- MThe Compiler Natively WASM execution

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

2022.12

v0.5
- Compiler Decorator Extension
- Policy & Flow Capability Enhancement
- Model Registry & Package Management
- More LSP Based IDEs
- Common Domain Language Programming Framework : Compiler-Base Stage 1

v0.8
- **Konfig (Model):** Support Aliyun ACK, ASM, Prometheus
- **Konfig (Toolbox):** Structure validation
- **Kusion (Resource):** Customimze resource health check
- **Security :** KCL Secret Management
- **IDE:** Kusion Operations Integration

2023.3

v0.6
- Lang Simplification Stage 2
- Reverse type inference
- Incremental compilation
- Multi Runtime/Backend

v0.9
- **Konfig (Model):** Support AWS EKS, App Mesh, AMP
- **Konfig (Toolbox):** Dependency analysis
- **Kusion (Operation):** Advanced workflow
- **Security:** Third-party KMS integration

2023.6

v0.7
- CFG-Based KCL IR
- Garbage collector
- JIT Compiler
- Compiler-Base Stage 2

v0.10
- **Konfig (Model):** Support Aliyun ECS, SLB, RDS
- **Konfig (Toolbox):** Pipeline Notification
- **Kusion (Operation):** Progressive rollout
- **Kusion (Operation):** Login identity
- **Kusion (Operation):** Pre/Post Hook
- **Kusion (Operation):** Operation REST

Kusion & Konfig

Resources

Fork me on GitHub

- Web Site
  - https://kusionstack.io/
- Source Code
  - https://github.com/KusionStack/kusion
  - https://github.com/KusionStack/KCLVM
  - https://github.com/KusionStack/konfig
- Contact
  - https://github.com/KusionStack/community#contact
  - https://github.com/KusionStack/community
- Twitter
  - @KusionStack

Thank you

KusionStack Team