KCL 在工程化配置策略场景的探索和落地使用

徐鹏飞

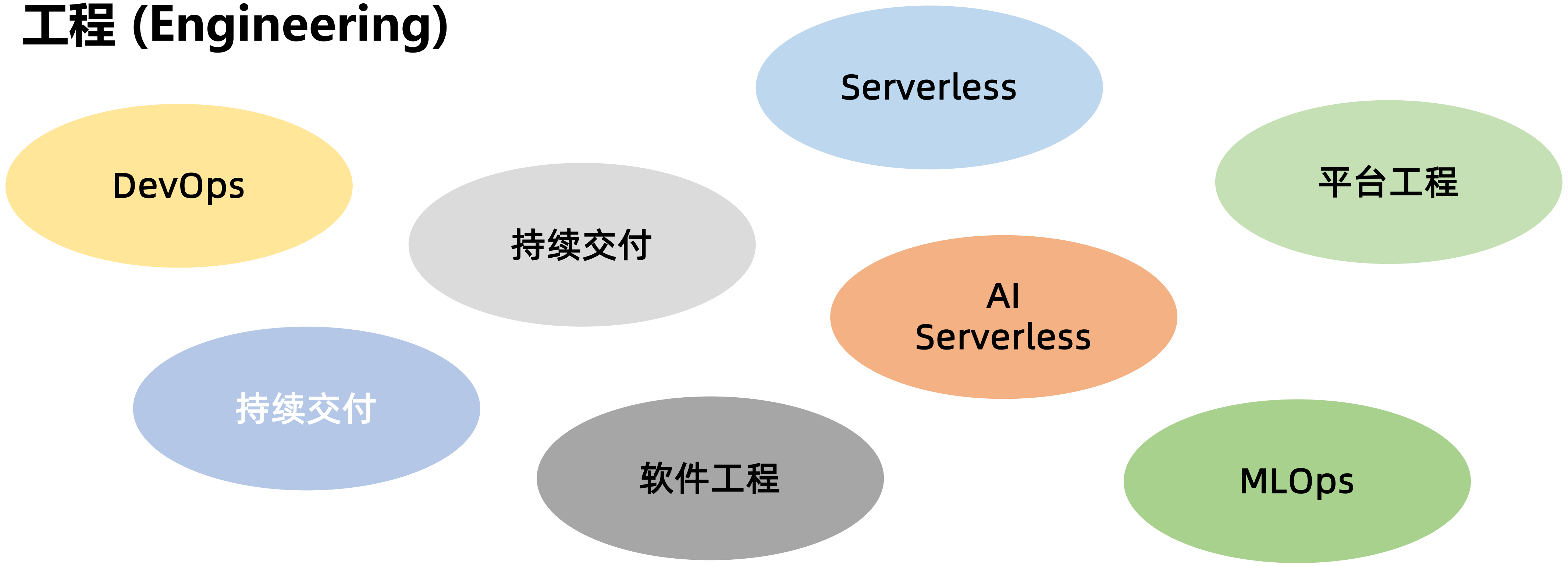
提纲

引言

01

引言

工程 (Engineering)

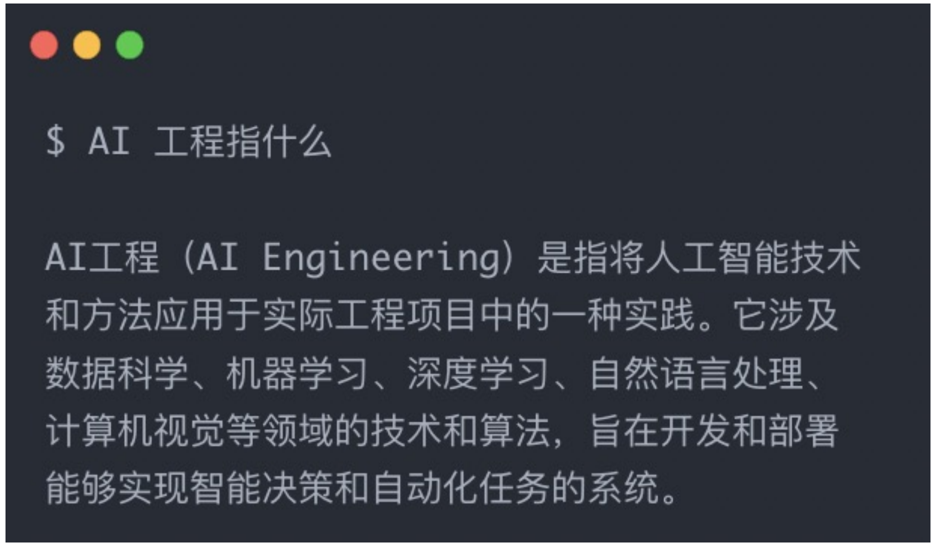


在 Gartner 公司发布的《2022年十二大重要战略技术趋势》中，将 **AI 工程（AI Engineering）** 列为未来三到五年企业数字业务创新的加速器。

在 Gartner 发布的《2023 年十大重要战略技术趋势》报告中，**平台工程 (Platform Engineering)** 位列其中。

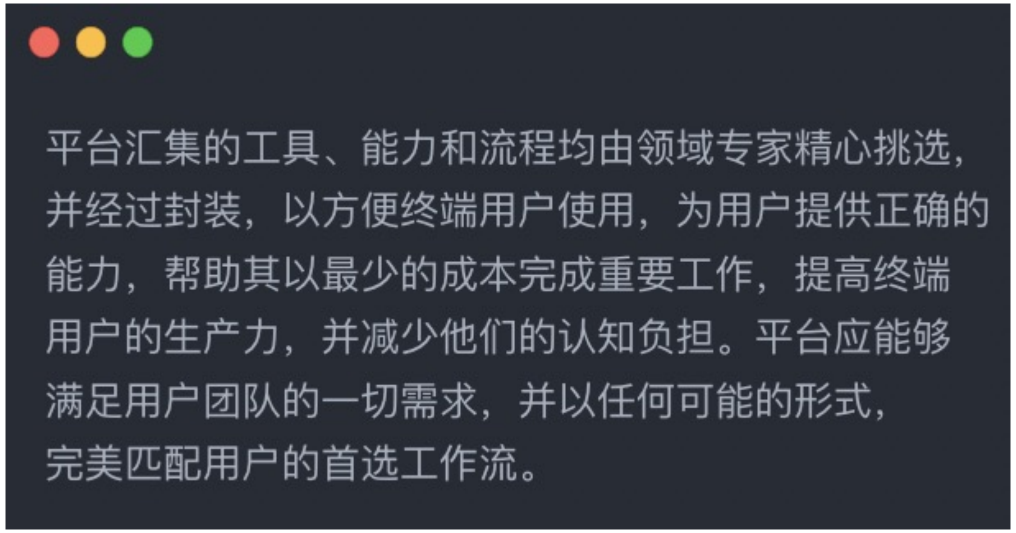
引言

平台工程 & AI 工程



```
$ AI 工程指什么

AI工程（AI Engineering）是指将人工智能技术
和方法应用于实际工程项目中的一种实践。它涉及
数据科学、机器学习、深度学习、自然语言处理、
计算机视觉等领域的技术和算法，旨在开发和部署
能够实现智能决策和自动化任务的系统。
```

让我们来看看 GPT 怎么说的?



```
平台汇集的工具、能力和流程均由领域专家精心挑选，
并经过封装，以方便终端用户使用，为用户提供正确的
能力，帮助其以最少的成本完成重要工作，提高终端
用户的生产力，并减少他们的认知负担。平台应能够
满足用户团队的一切需求，并以任何可能的形式，
完美匹配用户的首选工作流。
```

人工智能三要素：数据、算法、算力

数据是 AI 发展的重要基石，是不可或缺的资产和**训练样本**

我理解的工程化：方法、工具和实践的集合 - 体验至上，系统化，自动化，智能化，安全规范，可度量

引言

问题和挑战

认知负担

- 应用开发/ML 工程师/数据工程师/科学家需要面对复杂的基础设施和平台概念

配置/数据种类繁杂

- 结构化数据：其中表格数据类型是常见的数据表示形式
- 非结构化数据：文本，图像，视频，音频等

配置/数据清洗过程易出错

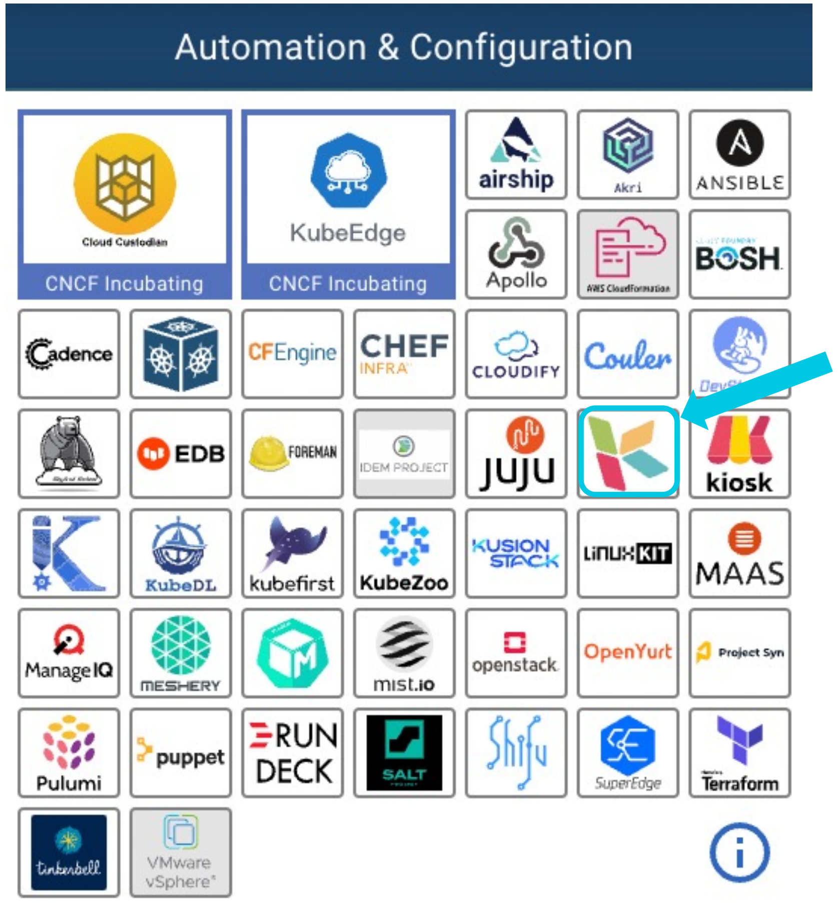
- 配置/数据值缺失、类型转换、异常值处理、合并/拆分过程缺乏标准且高效的结构定义和约束手段
- 配置/数据集种类繁杂，且喂到很多训练系统中需要做格式转换，缺乏有效的自动化和验证手段

效率可靠性低

- 缺乏标准的测试验证手段，大多是胶水代码或者脚本的拼盘

引言

KCL 专用配置策略语言 (2022.6 开源，2023.9 成为 CNCF 基金会托管的 Sandbox 项目)

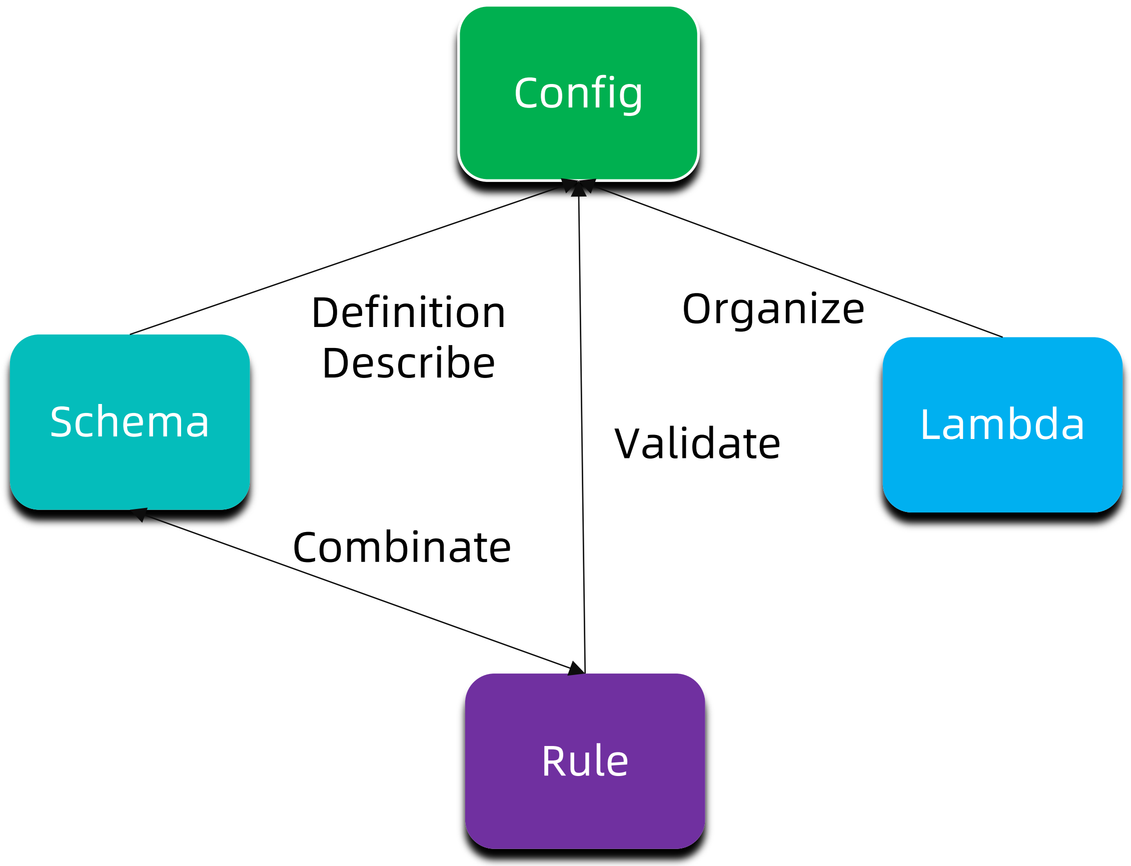


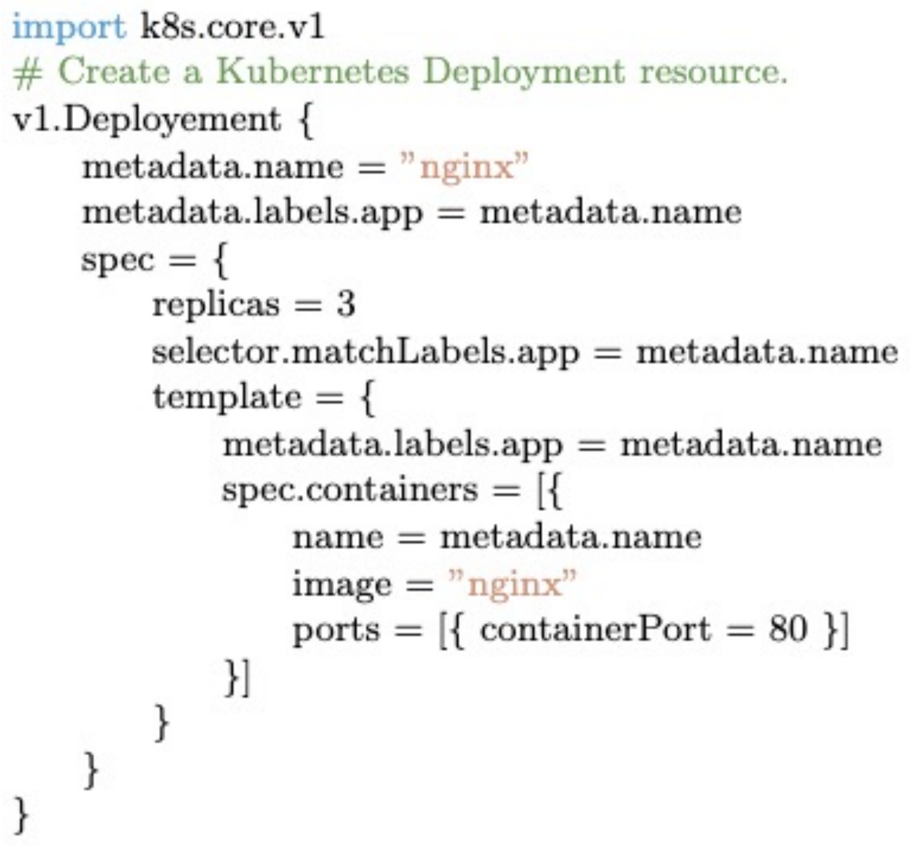
- ✓ **领域特定**：以收敛的语言和工具集合解决领域问题近乎无限的变化和复杂性，同时兼顾表达力和易用性
- ✓ **以数据和模型为中心**：开发者可以理解的声明式 Schema/配置/策略模型用于 AI 工程，云原生工程等场景
- ✓ **包含结构化定义和约束的核心数据结构 Schema**：为 AI 数据集或者云原生配置场景提供**原生的数据验证和转换**能力
- ✓ **可复用扩展**: OCI 等标准软件供应链集成和包管理工具支持，官方 Registry 提供 200+ 模型包
- ✓ **引擎解耦**：建立在一个完全开放的世界当中，几乎不与任何编排/引擎工具或者控制器绑定，可同时为客户端和运行时场景提供 API 抽象、组合和校验的能力
- ✓ **多语言 SDK：轻易集成到不同的业务场景和生态当中，目前提供了 Rust, Go, Python, Java 等 SDK**

引言

Config + Schema + Rule + Lambda

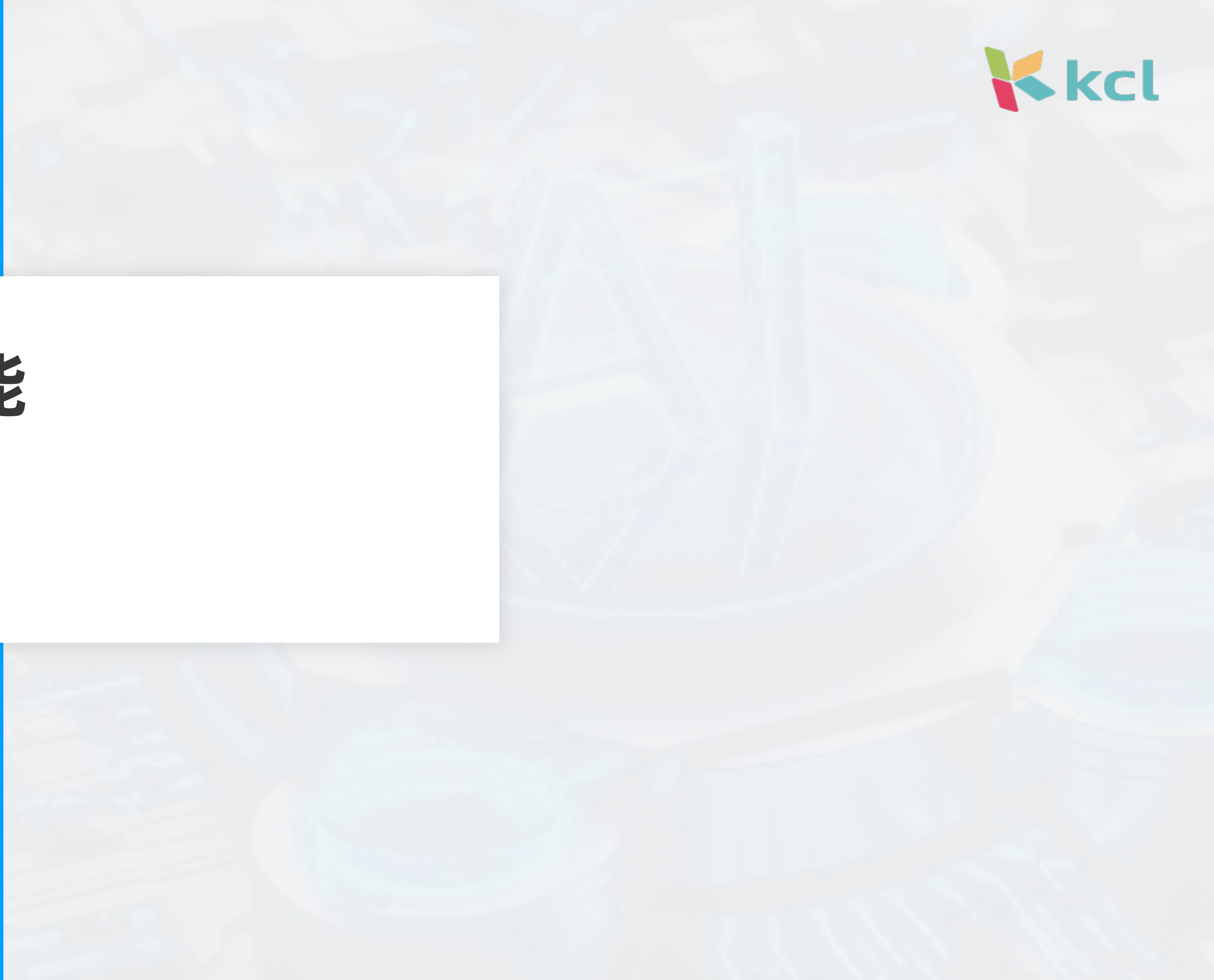
Pattern: $k = (T)v$





```
import k8s.core.v1
# Create a Kubernetes Deployment resource.
v1.Deployement {
    metadata.name = "nginx"
    metadata.labels.app = metadata.name
    spec = {
        replicas = 3
        selector.matchLabels.app = metadata.name
        template = {
            metadata.labels.app = metadata.name
            spec.containers = [{
                name = metadata.name
                image = "nginx"
                ports = [{ containerPort = 80 }]
            }]
        }
    }
}
```

可复用可扩展、抽象和组合能力、稳定性、高性能



场景功能

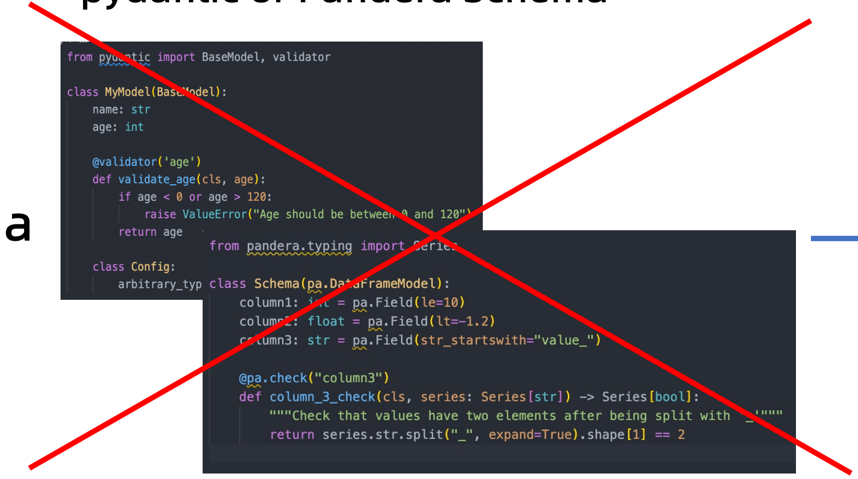
02

场景

表格数据集验证和转换

Schema

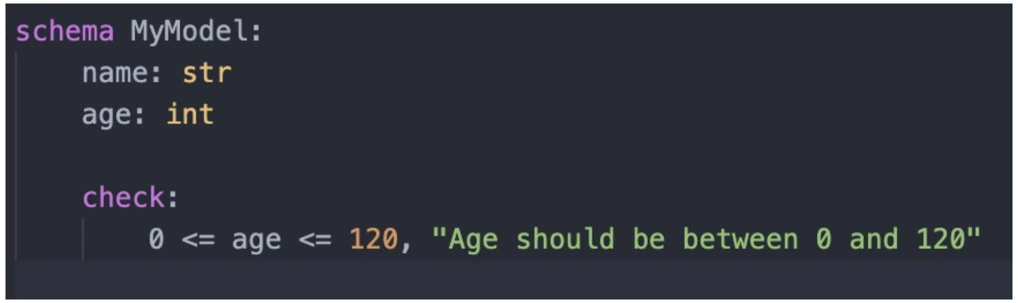
pydantic or Pandera Schema



```python
from pydantic import BaseModel, validator

class MyModel(BaseModel):
    name: str
    age: int

    @validator('age')
    def validate_age(cls, age):
        if age < 0 or age > 120:
            raise ValueError("Age should be between 0 and 120")
        return age

    class Config:
        arbitrary_typ
```

```python
from pandera.typing import Series

class Schema(pa.DataFrameModel):
    column1: int = pa.Field(le=10)
    column2: float = pa.Field(lt=-1.2)
    column3: str = pa.Field(str_startswith="value_")

    @pa.check("column3")
    def column_3_check(cls, series: Series[str]) -> Series[bool]:
        """Check that values have two elements after being split with '_'"""
        return series.str.split("_", expand=True).shape[1] == 2
```

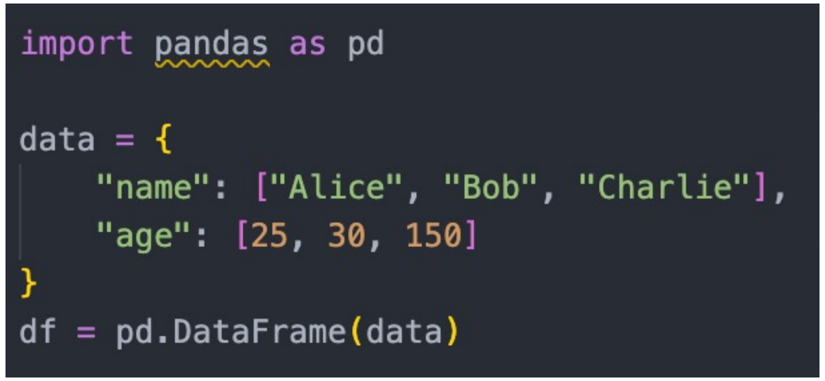
KCL Schema



```
schema MyModel:
    name: str
    age: int

    check:
        0 <= age <= 120, "Age should be between 0 and 120"
```

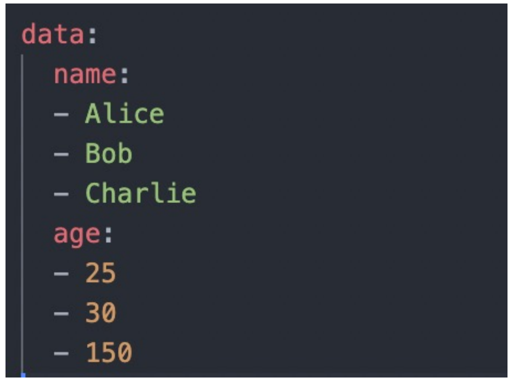
Data

DataFrame



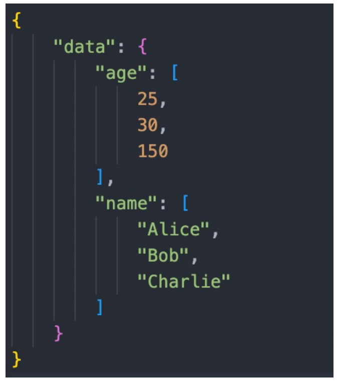
```python
import pandas as pd

data = {
    "name": ["Alice", "Bob", "Charlie"],
    "age": [25, 30, 150]
}
df = pd.DataFrame(data)
```

YAML



```yaml
data:
  name:
  - Alice
  - Bob
  - Charlie
  age:
  - 25
  - 30
  - 150
```

JSON

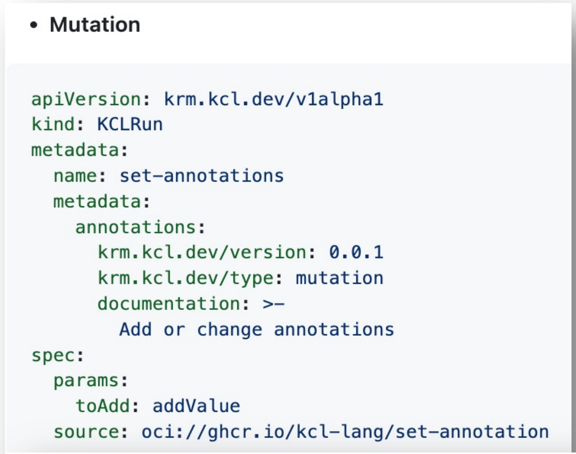


```json
{
  "data": {
    "age": [
      25,
      30,
      150
    ],
    "name": [
      "Alice",
      "Bob",
      "Charlie"
    ]
  }
}
```

平替复杂的 Schema 定义格式，类 Python 语法，简洁易读，多种数据验证支持，多语言 SDK 支持

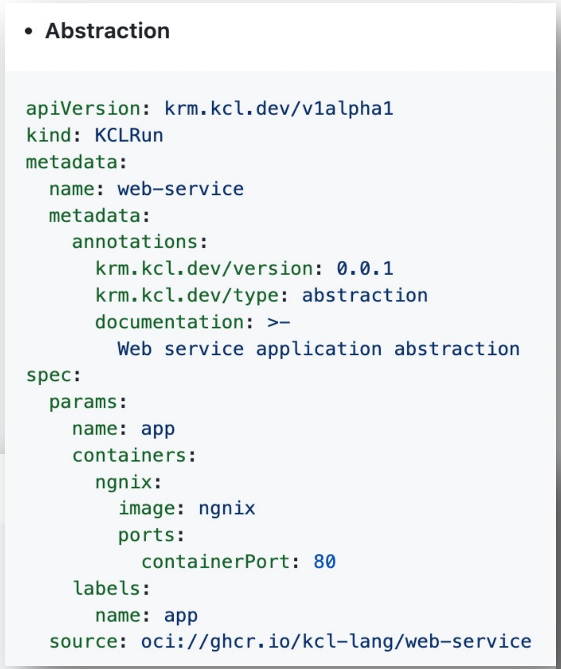
场景

云原生配置验证和转换



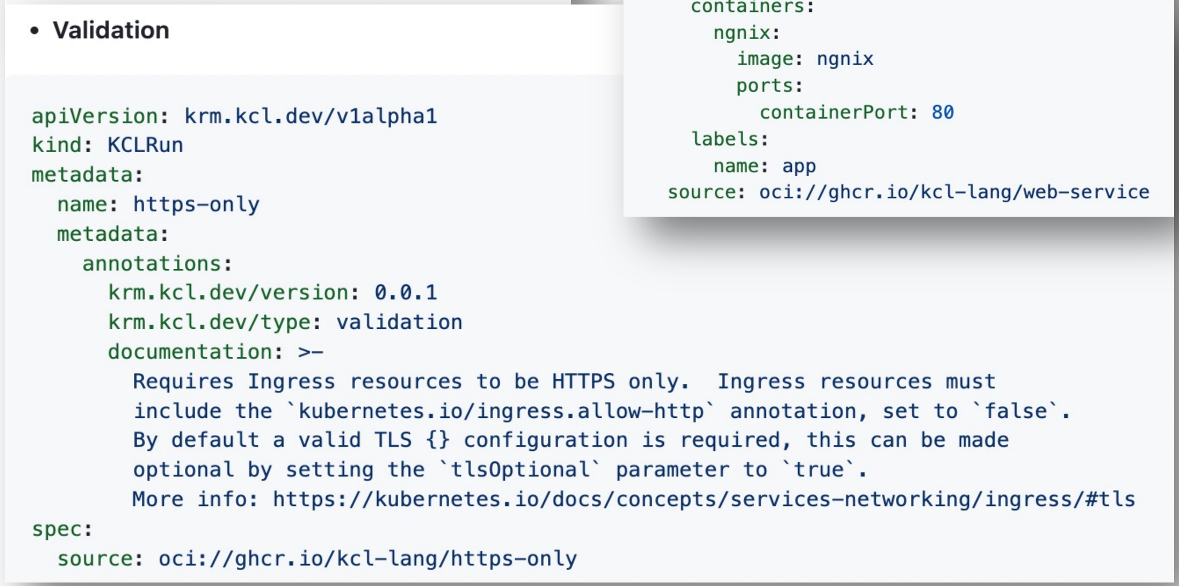
- Mutation

```
apiVersion: krm.kcl.dev/v1alpha1
kind: KCLRun
metadata:
  name: set-annotations
  metadata:
    annotations:
      krm.kcl.dev/version: 0.0.1
      krm.kcl.dev/type: mutation
      documentation: >-
        Add or change annotations
spec:
  params:
    toAdd: addValue
  source: oci://ghcr.io/kcl-lang/set-annotation
```



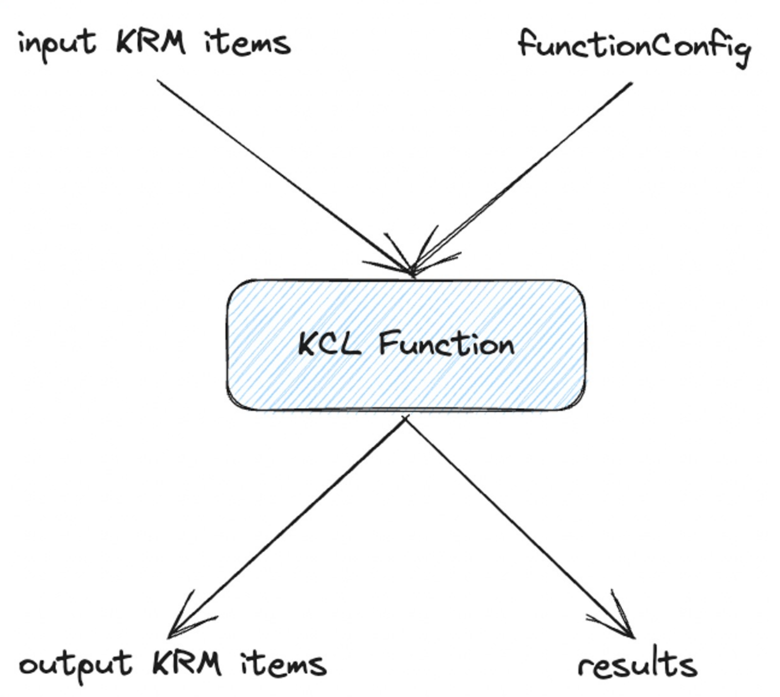
- Abstraction

```
apiVersion: krm.kcl.dev/v1alpha1
kind: KCLRun
metadata:
  name: web-service
  metadata:
    annotations:
      krm.kcl.dev/version: 0.0.1
      krm.kcl.dev/type: abstraction
      documentation: >-
        Web service application abstraction
spec:
  params:
    name: app
    containers:
      ngnix:
        image: ngnix
        ports:
          containerPort: 80
    labels:
      name: app
  source: oci://ghcr.io/kcl-lang/web-service
```



- Validation

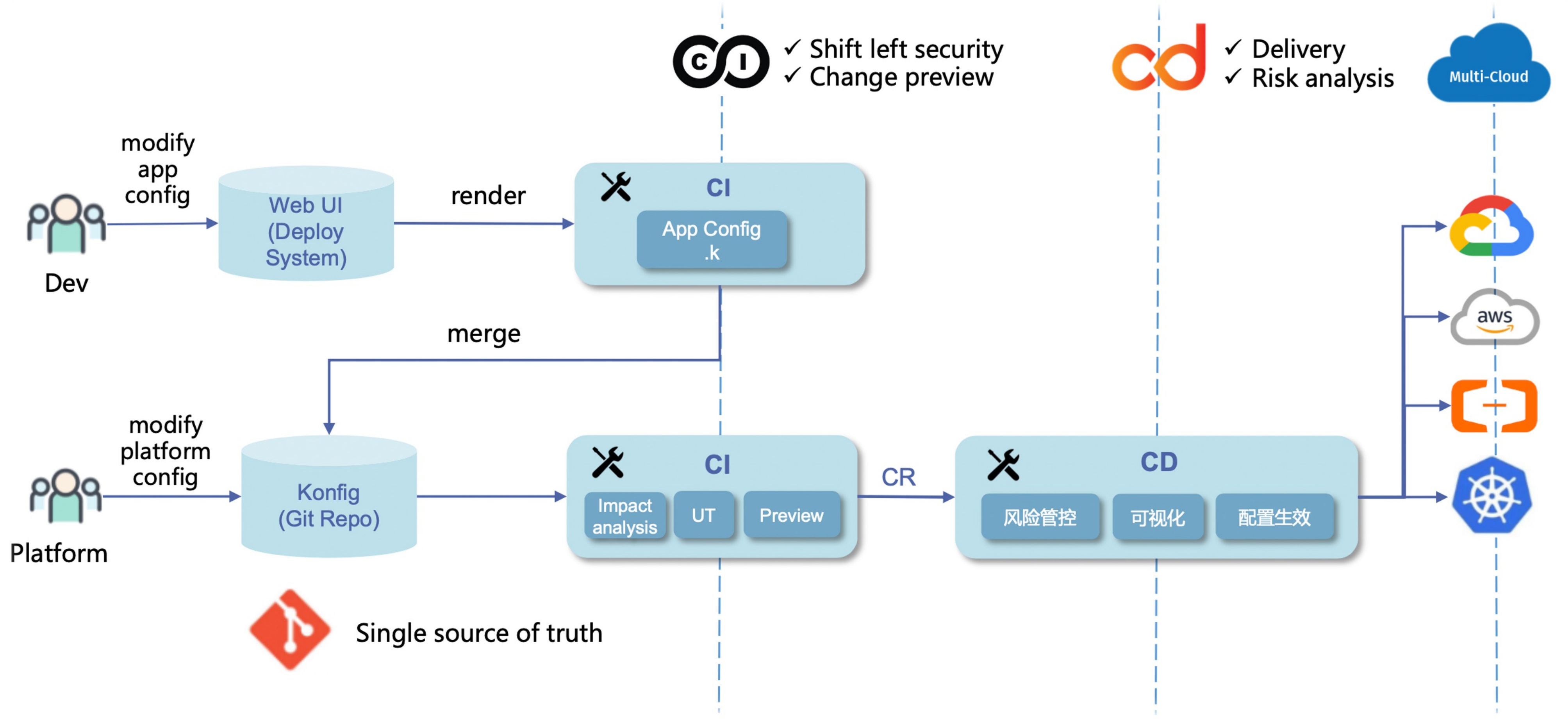
```
apiVersion: krm.kcl.dev/v1alpha1
kind: KCLRun
metadata:
  name: https-only
  metadata:
    annotations:
      krm.kcl.dev/version: 0.0.1
      krm.kcl.dev/type: validation
      documentation: >-
        Requires Ingress resources to be HTTPS only.  Ingress resources must
        include the `kubernetes.io/ingress.allow-http` annotation, set to `false`.
        By default a valid TLS {} configuration is required, this can be made
        optional by setting the `tlsOptional` parameter to `true`.
        More info: https://kubernetes.io/docs/concepts/services-networking/ingress/#tls
spec:
  source: oci://ghcr.io/kcl-lang/https-only
```



- 遵循统一的 KRM Function 规范
- 多种代码源支持: OCI, Git, Https, Filesystem…
- 可编程可扩展: 使用 KCL 语言简单编写模型

场景

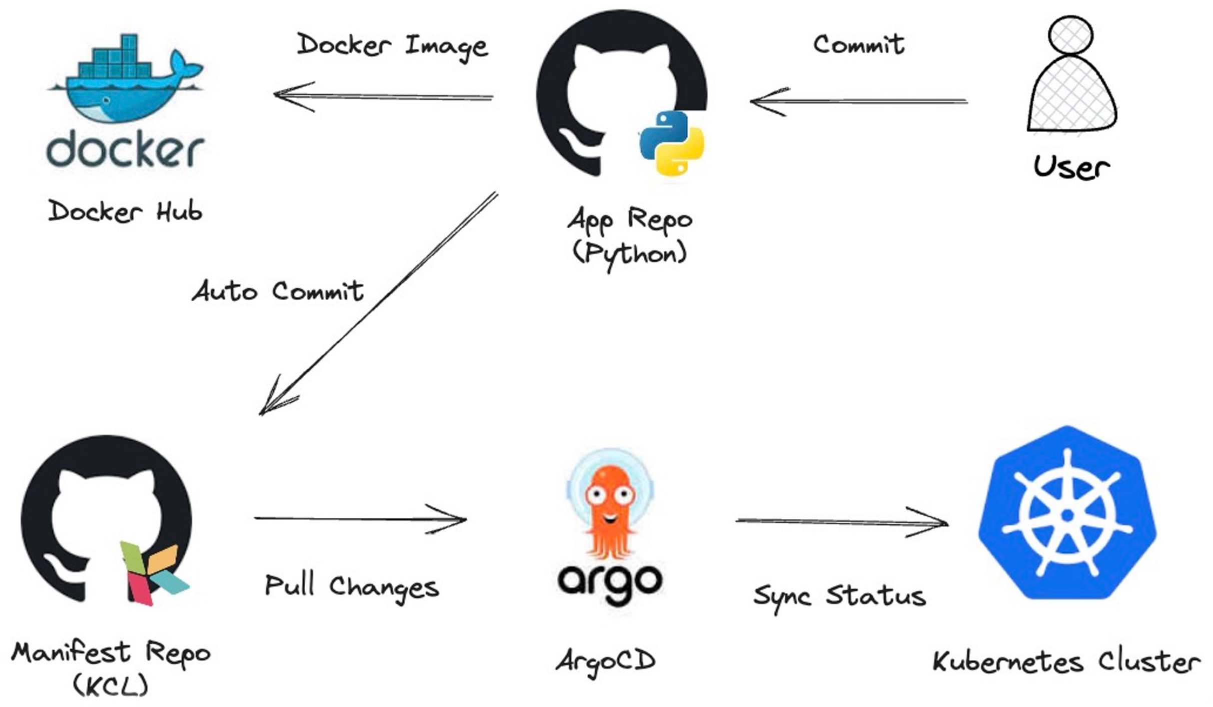
通过抽象进行应用交付

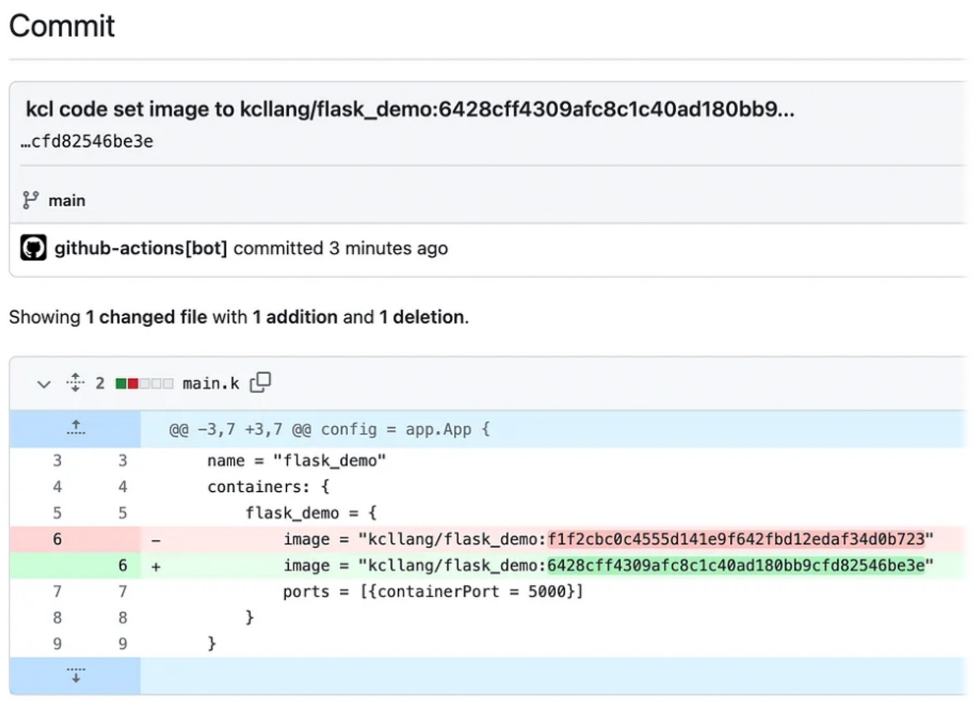


多种交付引擎支持：KusionStack, Kubevela, ...

场景

IaC & GitOps




Commit

kcl code set image to kcllang/flask_demo:6428cff4309afc8c1c40ad180bb9...
…cfd82546be3e

main

github-actions[bot] committed 3 minutes ago

Showing **1 changed file** with **1 addition** and **1 deletion**.

2 main.k

```diff
@@ -3,7 +3,7 @@ config = app.App {
     name = "flask_demo"
     containers: {
         flask_demo = {
-            image = "kcllang/flask_demo:f1f2cbc0c4555d141e9f642fbd12edaf34d0b723"
+            image = "kcllang/flask_demo:6428cff4309afc8c1c40ad180bb9cfd82546be3e"
             ports = [{containerPort = 5000}]
         }
     }
```

配置驱动的工作流：多种 CI/CD 和 GitOps 工具支持 e.g., GitHub Action, ArgoCD

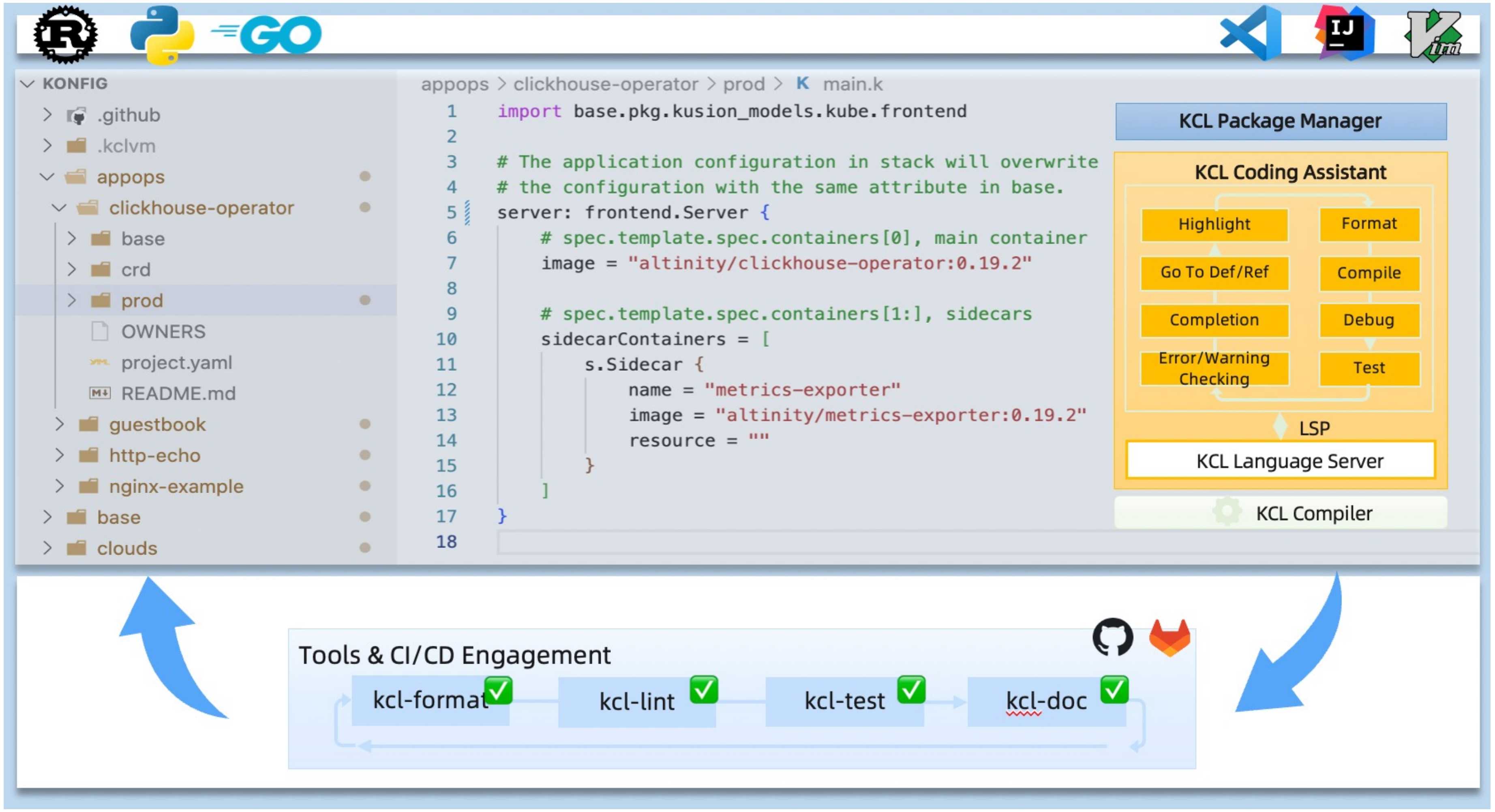
https://kcl-lang.io/docs/user_docs/guides/gitops/gitops-quick-start

开发者体验

03

IDE & 工具链

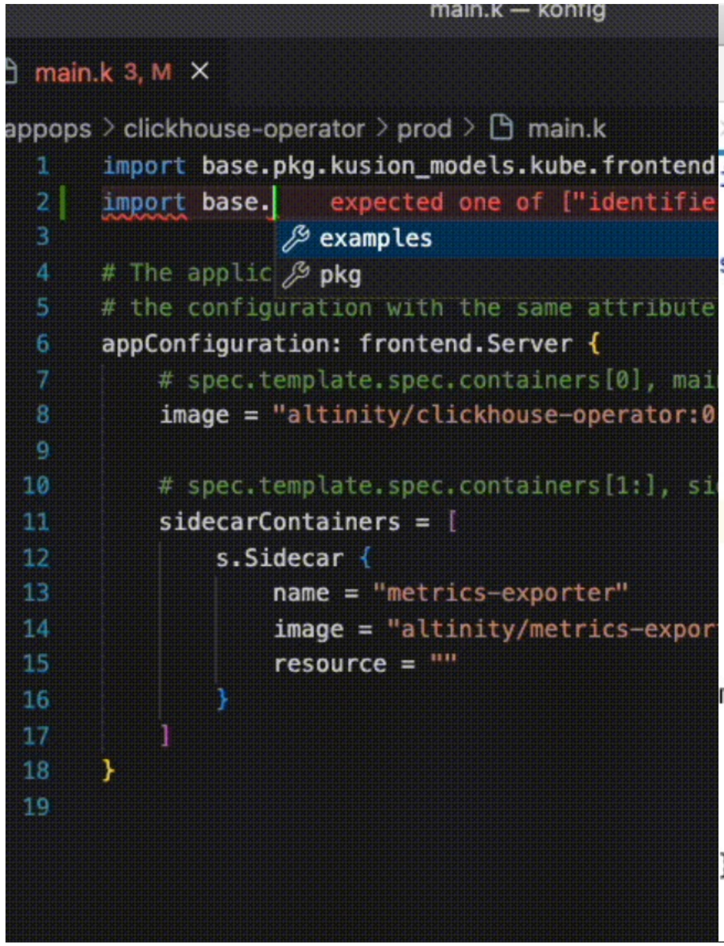
Language + Tools + IDEs + SDKs + Plugins



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

# The application configuration in stack will overwrite
# the configuration with the same attribute in base.
server: frontend.Server {
    # spec.template.spec.containers[0], main container
    image = "altinity/clickhouse-operator:0.19.2"

    # spec.template.spec.containers[1:], sidecars
    sidecarContainers = [
        s.Sidecar {
            name = "metrics-exporter"
            image = "altinity/metrics-exporter:0.19.2"
            resource = ""
        }
    ]
}
```

IDE & 工具链

- VS Code
- Idea
- NeoVim

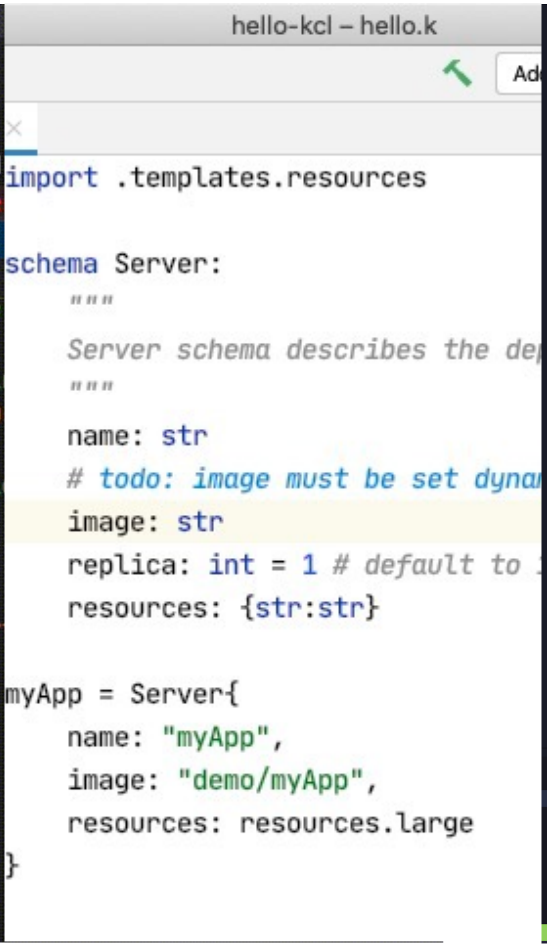


main.k — konfig

main.k 3, M ×

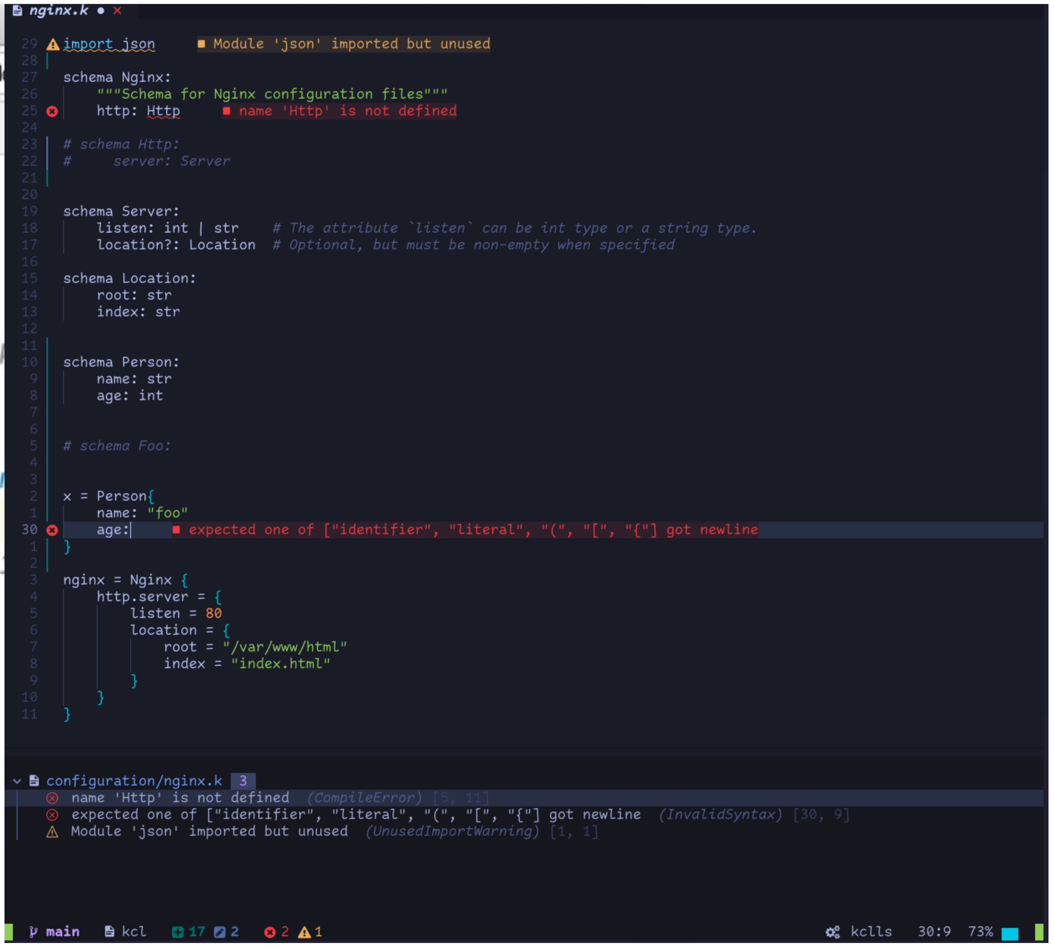
appops > clickhouse-operator > prod > main.k

```
import base.pkg.kusion_models.kube.frontend
import base.|    expected one of ["identifie
                 examples
# The applic     pkg
# the configuration with the same attribute
appConfiguration: frontend.Server {
    # spec.template.spec.containers[0], mai
    image = "altinity/clickhouse-operator:0

    # spec.template.spec.containers[1:], si
    sidecarContainers = [
        s.Sidecar {
            name = "metrics-exporter"
            image = "altinity/metrics-expor
            resource = ""
        }
    ]
}
```



hello-kcl – hello.k

```
import .templates.resources

schema Server:
    """
    Server schema describes the de
    """
    name: str
    # todo: image must be set dyna
    image: str
    replica: int = 1 # default to
    resources: {str:str}

myApp = Server{
    name: "myApp",
    image: "demo/myApp",
    resources: resources.large
}
```



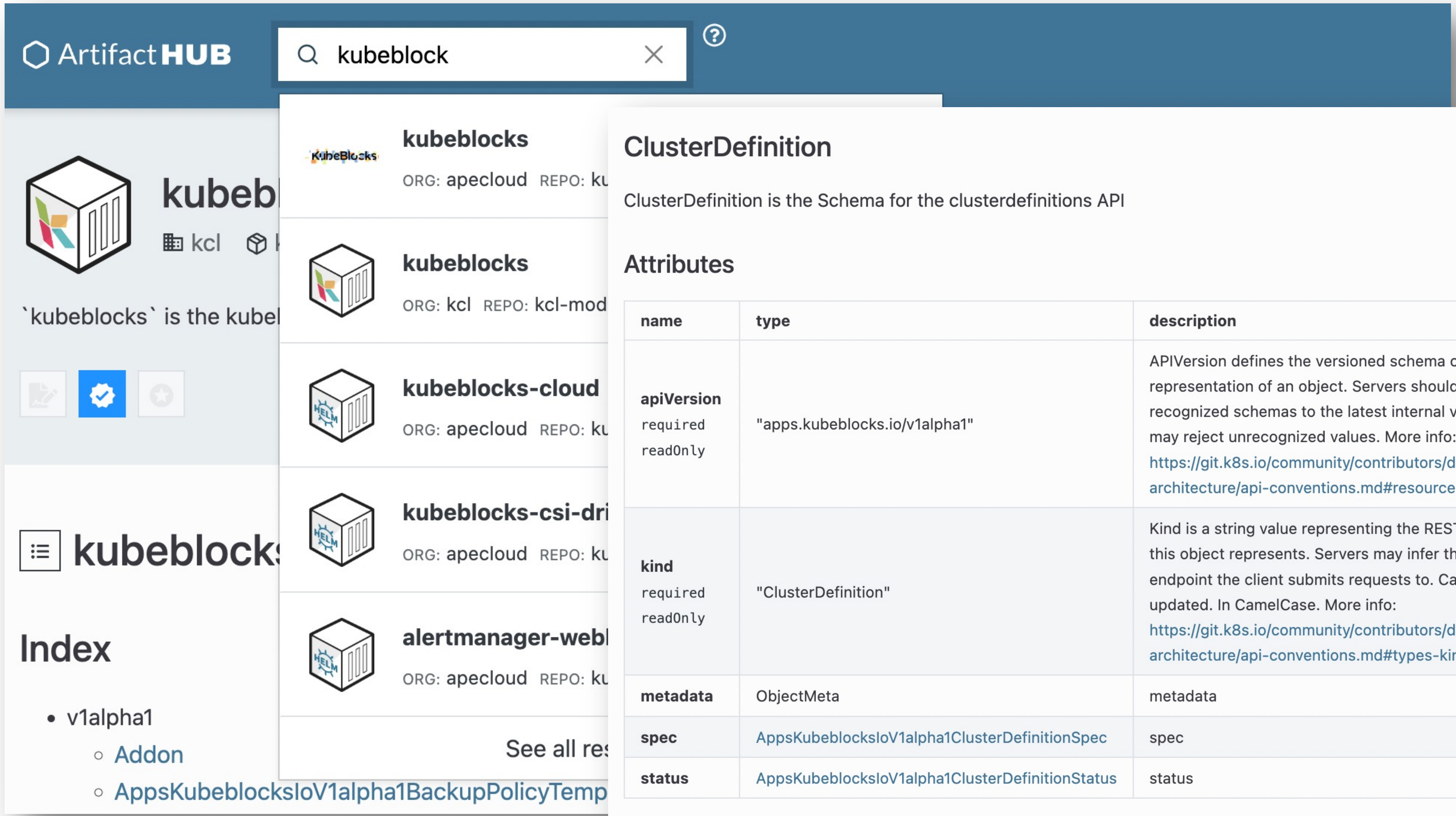
nginx.k ● ×

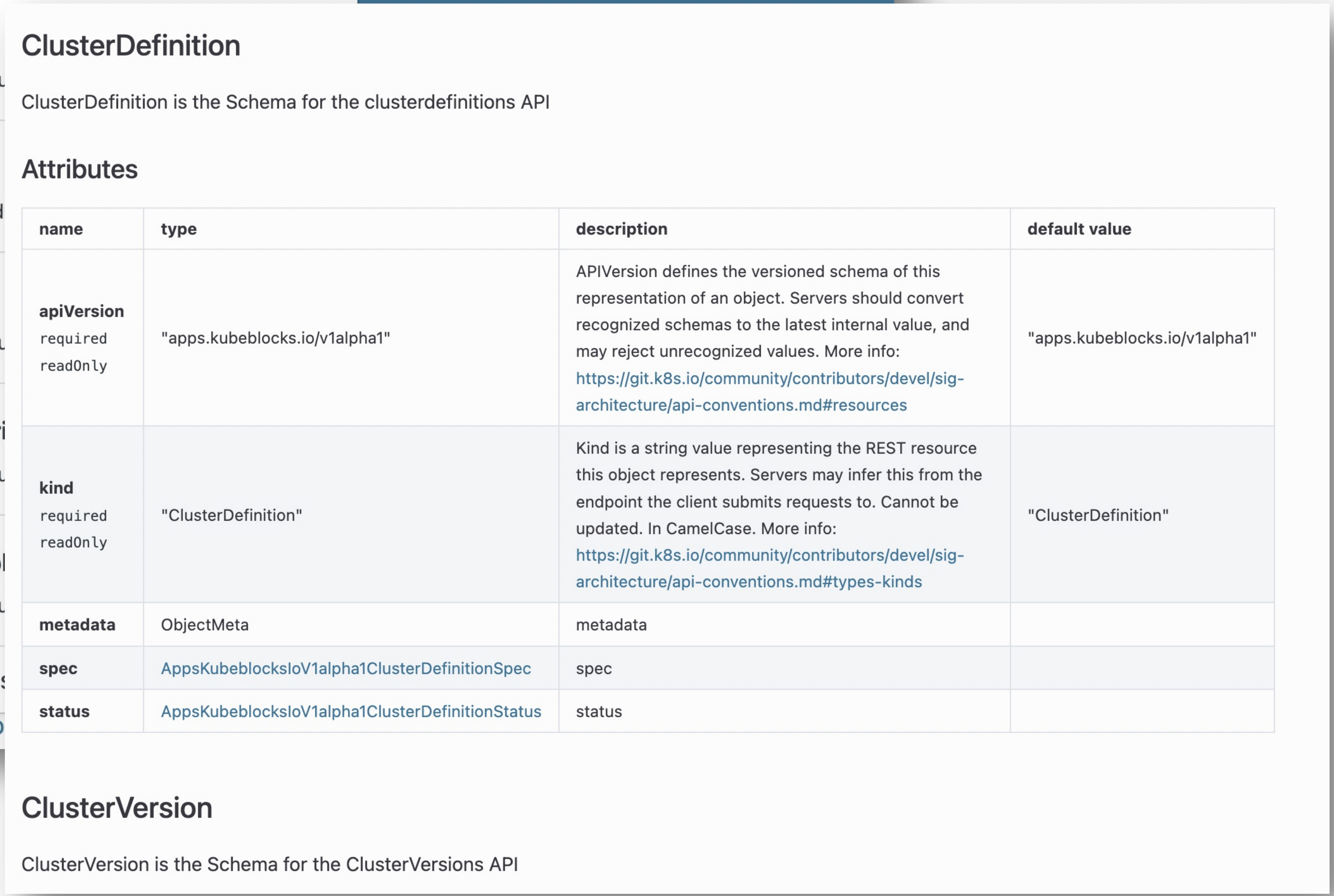
```
import json     ■ Module 'json' imported but unused

schema Nginx:
    """Schema for Nginx configuration files"""
    http: Http      ■ name 'Http' is not defined

# schema Http:
#     server: Server


schema Server:
    listen: int | str    # The attribute `listen` can be int type or a string type.
    location?: Location  # Optional, but must be non-empty when specified

schema Location:
    root: str
    index: str


schema Person:
    name: str
    age: int


# schema Foo:


x = Person{
    name: "foo"
    age:|      ■ expected one of ["identifier", "literal", "(", "[", "{"] got newline
}

nginx = Nginx {
    http.server = {
        listen = 80
        location = {
            root = "/var/www/html"
            index = "index.html"
        }
    }
}
```

```
configuration/nginx.k  3
  ⊗ name 'Http' is not defined  (CompileError) [5, 11]
  ⊗ expected one of ["identifier", "literal", "(", "[", "{"] got newline  (InvalidSyntax) [30, 9]
  ⚠ Module 'json' imported but unused  (UnusedImportWarning) [1, 1]
```

```
main   kcl   17  2   2  1                                         kclls   30:9   73%
```

模型 Registry





ClusterDefinition

ClusterDefinition is the Schema for the clusterdefinitions API

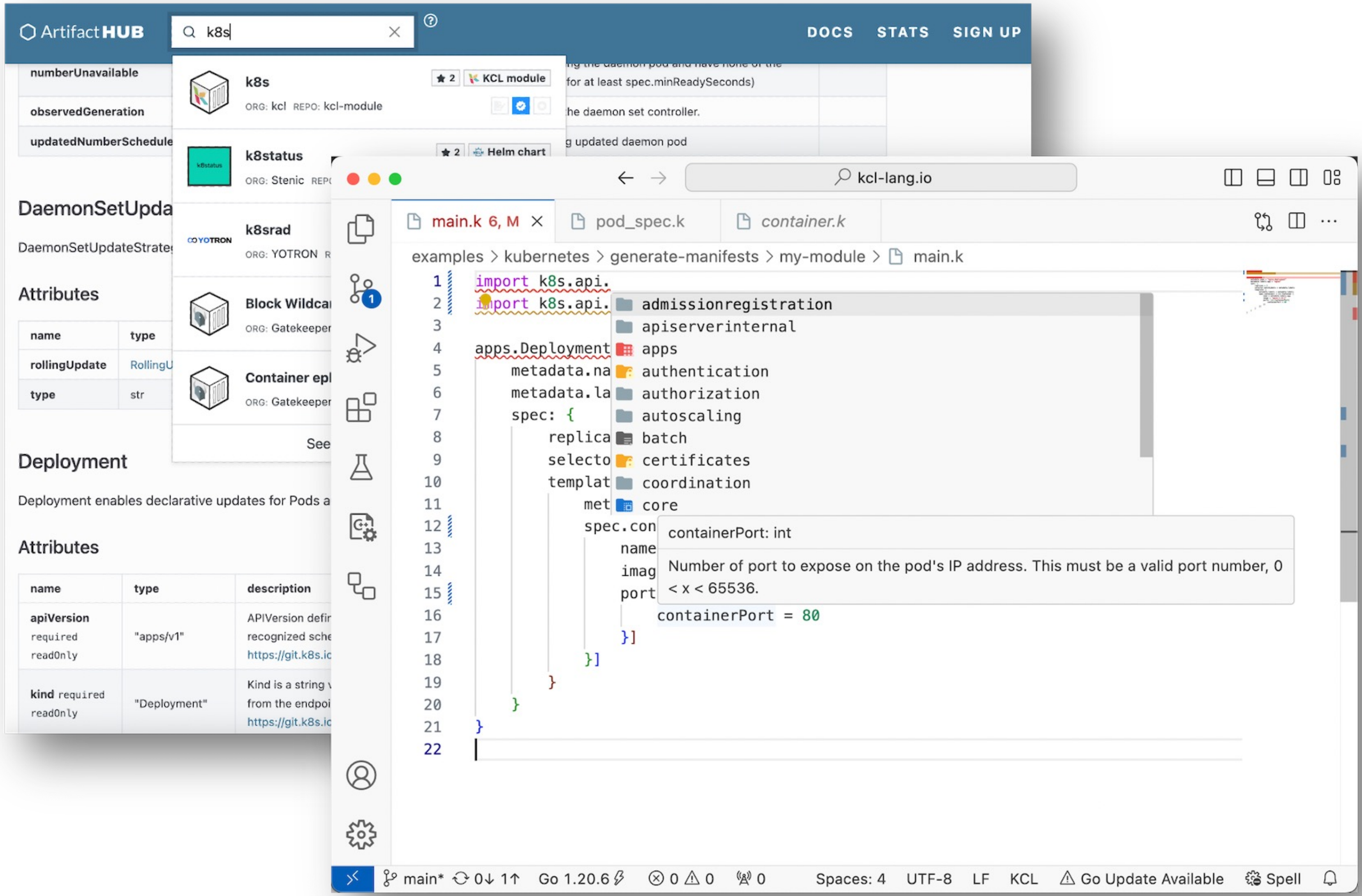
Attributes

name	type	description	default value
apiVersion `required` `readOnly`	"apps.kubeblocks.io/v1alpha1"	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources	"apps.kubeblocks.io/v1alpha1"
kind `required` `readOnly`	"ClusterDefinition"	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds	"ClusterDefinition"
metadata	ObjectMeta	metadata	
spec	AppsKubeblocksIoV1alpha1ClusterDefinitionSpec	spec	
status	AppsKubeblocksIoV1alpha1ClusterDefinitionStatus	status	

ClusterVersion

ClusterVersion is the Schema for the ClusterVersions API

模型 Registry



numberUnavailable
observedGeneration
updatedNumberSchedule

...ng the daemon pod and have none of the... for at least spec.minReadySeconds)
...he daemon set controller.
...g updated daemon pod

k8s ★ 2 KCL module
ORG: kcl REPO: kcl-module

k8status ★ 2 Helm chart
ORG: Stenic REPO

k8srad
ORG: YOTRON

Block Wildca
ORG: Gatekeeper

Container epl
ORG: Gatekeeper

See

DaemonSetUpda

DaemonSetUpdateStrate

Attributes

name	type
rollingUpdate	RollingU
type	str

Deployment

Deployment enables declarative updates for Pods a

Attributes

name	type	description
apiVersion required readOnly	"apps/v1"	APIVersion defir recognized sche https://git.k8s.io
kind required readOnly	"Deployment"	Kind is a string v from the endpoi https://git.k8s.io

kcl-lang.io

main.k 6, M | pod_spec.k | container.k

examples > kubernetes > generate-manifests > my-module > main.k

```
import k8s.api.
import k8s.api.

apps.Deployment
    metadata.na
    metadata.la
    spec: {
        replica
        selecto
        templat
            met
            spec.con
                name
                imag
                port
                    containerPort = 80
                }]
            }]
        }
    }
}
```

```
admissionregistration
apiserverinternal
apps
authentication
authorization
autoscaling
batch
certificates
coordination
core
```

containerPort: int

Number of port to expose on the pod's IP address. This must be a valid port number, 0 < x < 65536.

main* 0↓ 1↑ Go 1.20.6 ⊗ 0 ⚠ 0 0 Spaces: 4 UTF-8 LF KCL ⚠ Go Update Available Spell

生态集成

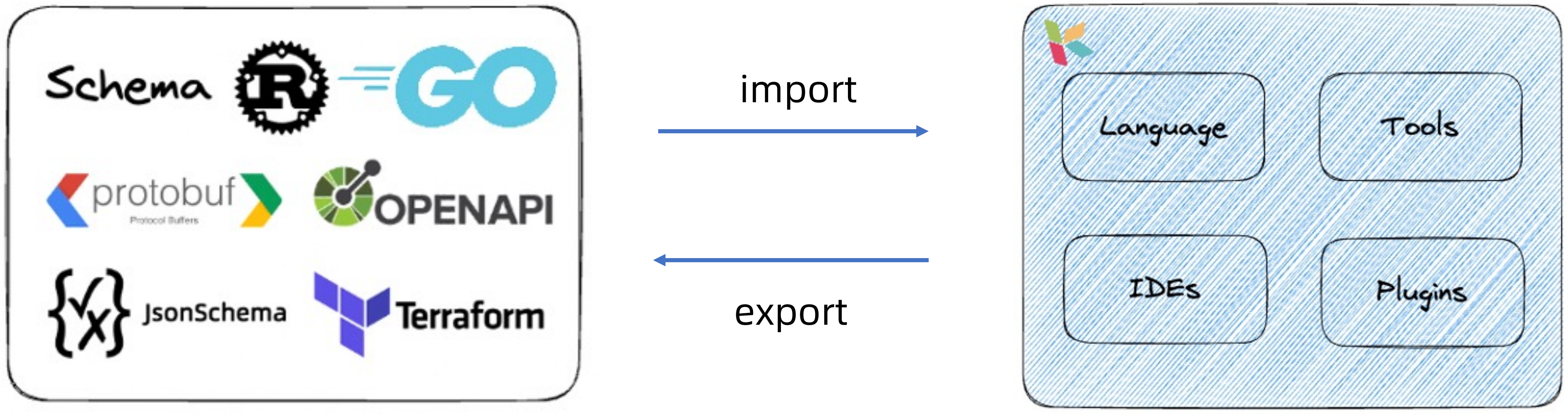
云原生工具集成

Integrate with Your Favorite Projects

- **KRM 支持**: 统一的规范和插件支持 e.g., kubectl-kcl plugin, helm-kcl plugin, helmfile-kcl plugin, kustomize-kcl plugin, kpt-kcl-plugin, crossplane kcl function, ...
- **运行时集成**: 使用 KCL Operator 而不是重复开发 Kubernetes Admission Webhook

生态集成

Schema 集成



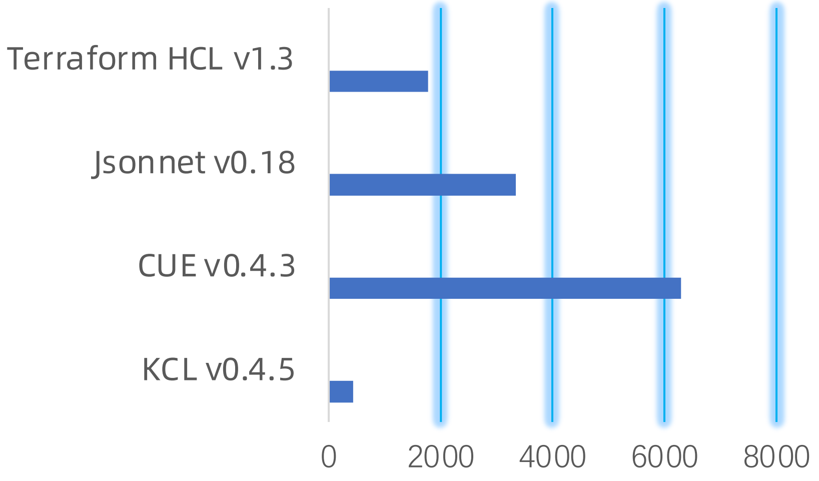
Python App (👷 施工中)

FastAPI Pydantic UNION PANDERA scikit learn

性能

Loop and Function

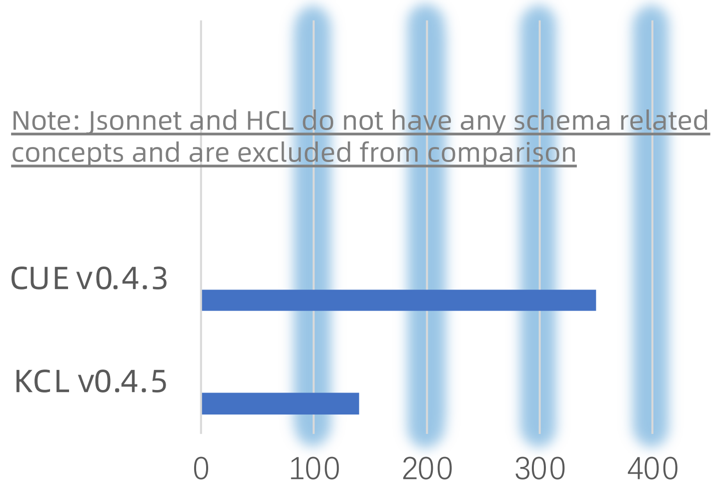
```
a = lambda x: int, y: int -> int {
    max([x, y])
}
temp = {"a${i}": a(1, 2) for i in range(10000)}
```



Kubernetes Configuration

```
import kubernetes.api.apps.v1

deployment = v1.Deployment {}
```

Note: Jsonnet and HCL do not have any schema related concepts and are excluded from comparison



Test environment: single core macOS 10.15.7 CPU: i7-8850H 2.6GHz 32GB 2400Mhz DDR4 No NUMA, e2e run time (ms)

总结

04

总结

Mutation, Validation, Abstraction
Production-Ready

KCL is an open-source constraint-based record & functional language mainly used in configuration and policy scenarios.

- 通过工程化方式提供合适配置/数据编辑、校验手段
- 通过定义合适的抽象隐藏基础设施和平台细节，减轻开发人员的负担。
- 通过通过更现代的声明式配置策略语言和工具，KRM KCL 规范, OCI Registry 和 Artifact Hub 等，帮助不同团队/角色之间更轻松地共享、传播和交付模型。（欢迎共建模型👏）

更多资源

- 官方网站
  - https://kcl-lang.io/
- GitHub
  - https://github.com/kcl-lang
- Twitter
  - @kcl language
- Slack
  - CNCF KCL Slack Channel: https://cloud-native.slack.com/archives/C05TC96NWN8

钉钉(DingTalk ID 42753001)

微信公众号

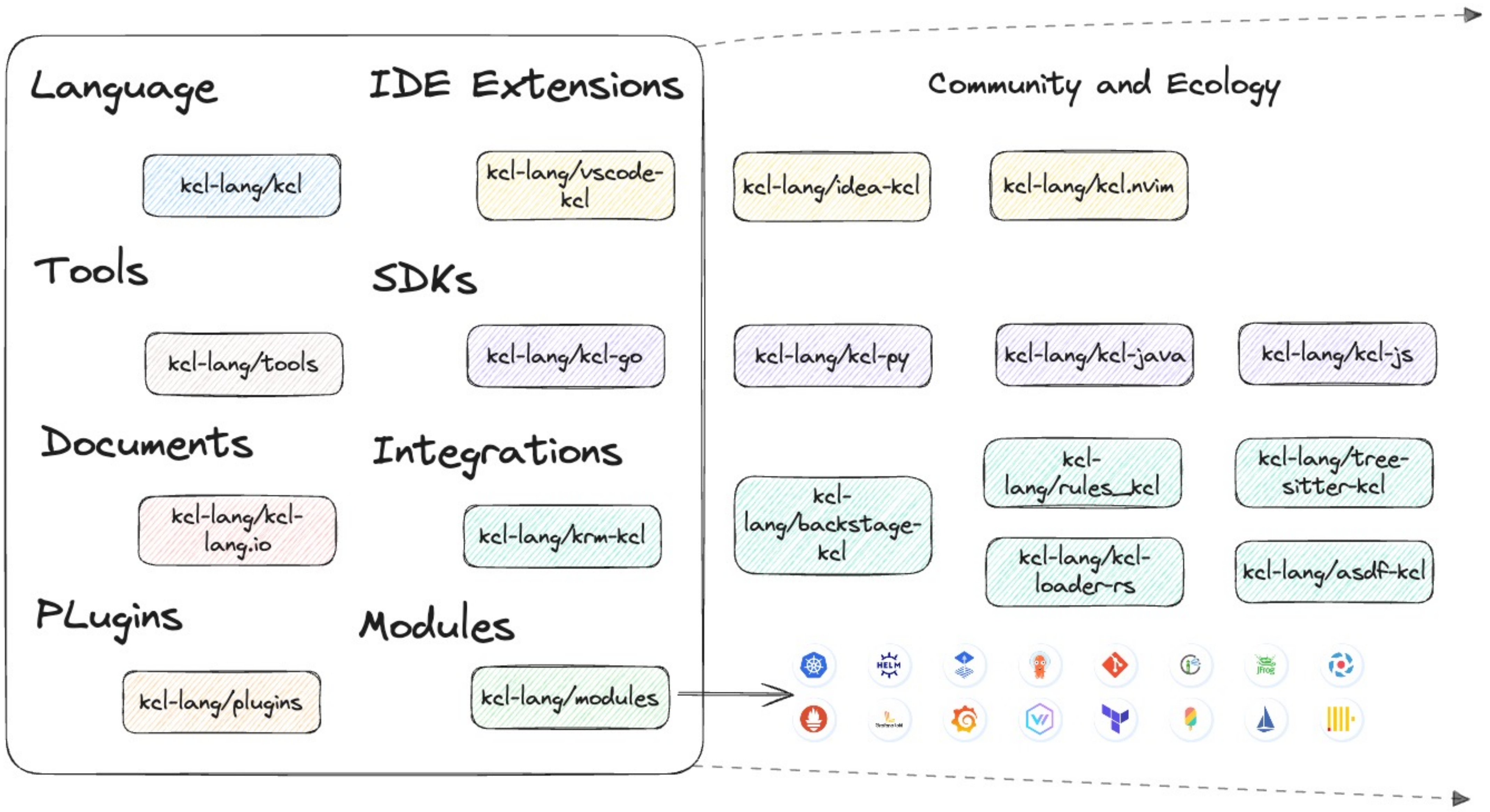
微信搜一搜

规模化云原生运维

附录

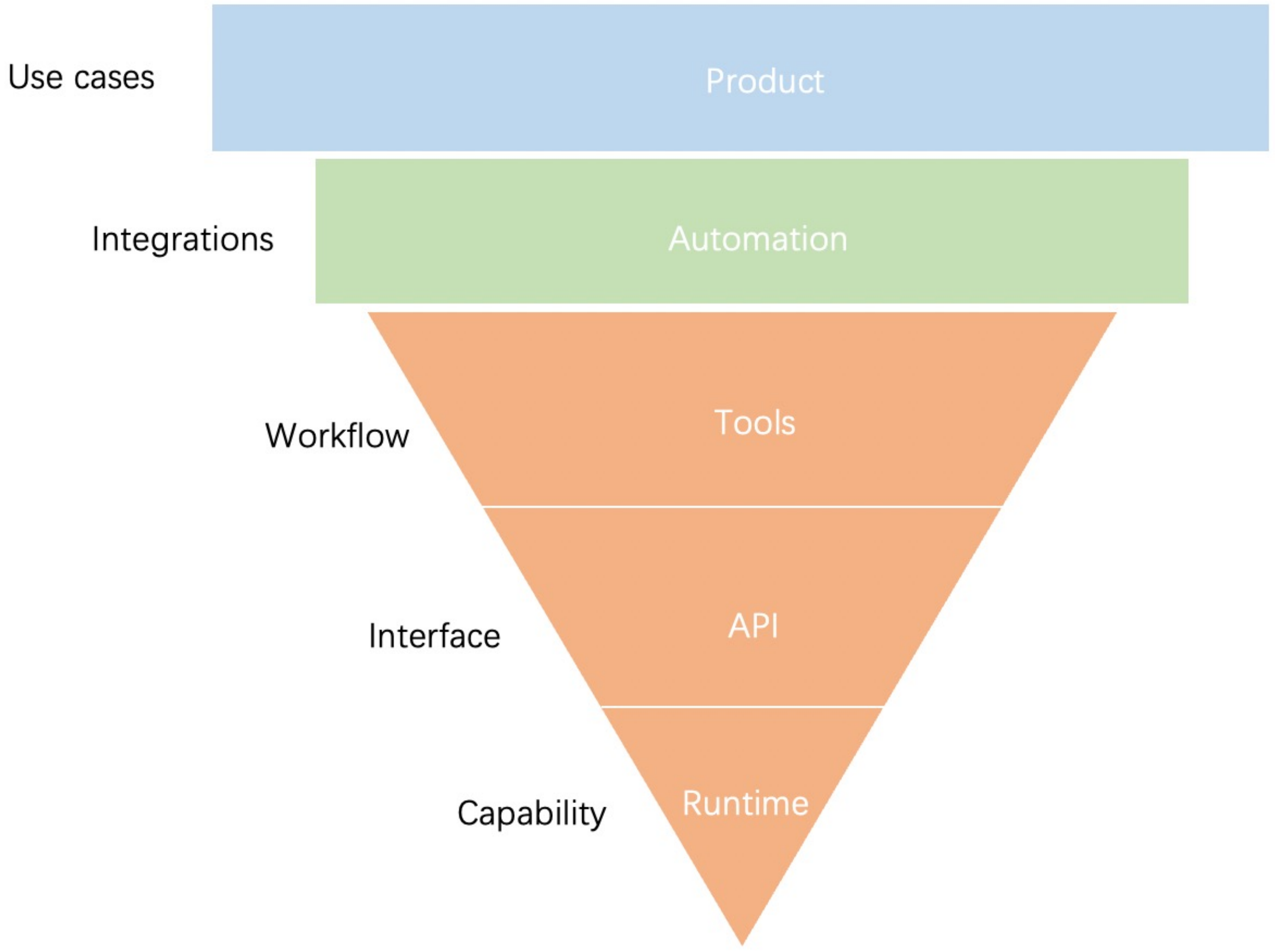
KCL 社区

KCL 项目组成



Language
kcl-lang/kcl
IDE Extensions
kcl-lang/vscode-kcl
Tools
kcl-lang/tools
SDKs
kcl-lang/kcl-go
Documents
kcl-lang/kcl-lang.io
Integrations
kcl-lang/krm-kcl
PLugins
kcl-lang/plugins
Modules
kcl-lang/modules
Community and Ecology
kcl-lang/idea-kcl
kcl-lang/kcl.nvim
kcl-lang/kcl-py
kcl-lang/kcl-java
kcl-lang/kcl-js
kcl-lang/backstage-kcl
kcl-lang/rules_kcl
kcl-lang/tree-sitter-kcl
kcl-lang/kcl-loader-rs
kcl-lang/asdf-kcl
HELM

KCL 技术组件



IaC & PaC Based KCL
GitOps & GitClickOps Based KCL
Dynamic Configuration & Package Management for Clouds and Kubernetes

Workflow Integrations (CI & CD Tools & DevOps Tools): KusionStack,
API Composition Tool Integrations: Crossplane, Kustomize, Helm, …
Hub & Registry Integration: Artifact Hub, Docker Hub, Harbor
Data & Schema Integration: Open API, Terraform, Kubernetes
Config UI: Backstage Plugin, Form Component Integrations

Tooling: run, mod, registry, vet, plugin, deps, doc, fmt, import, export, test, debug, fix, lint
IDEs: UI Extension & Client, Language Server: Definition, Reference, Diagnostic, Completion, …
SDKs: Rust, Go, Python, Java, …

API: API Server, Lang APIs, Plugin APIs, Tooling APIs, LSP APIs
Lang UI: Syntax, AST, Data Types.
Features: query, override, entry, top level arguments, error handling, …
Plugin: Multilingual or WASM function plugins

Sematic: Type system, sematic rules.
Performance: Parse, Resolve, Compile, Evaluation
Cross-platform: macOS, Linux, Windows, WASM
Runtime Capabilities: IR, Profiling, Debuging, System Modules, Memory management, Exception handling
Compiler Data Structure and Algorithms

社区项目对比

Structured KV

Pros.

- Easy to write and read
- Rich multi-language API
- Various Path Tools

Cons.

- Redundant information
- Insufficient functionality e.g. abstraction, constraint, …

Tech.

- JSON
- YAML

Product

- Kustomize
- …

Templated KV

Pros.

- Simple config logic support
- Dynamic argument input

Cons.

- Increase of argument makes it difficult to maintain
- Insufficient functionality e.g. abstraction, constraint, …

Tech.

- Velocity
- Go Template

Product

- Helm
- Helmfile
- …

Programmable KV

Pros.

- Required programming features
- Code modularity
- Templates & Data abstraction

Cons.

- Insufficient type constraints
- Insufficient restraint ability
- Runtime error

Tech.

- GCL
- HCL
- Bicep
- Starlark
- Jsonnet
- CEL
- OPA/Rego
- …

Product

- Terraform
- Tanka
- Radius
- ytt
- kpt

Typed KV

Pros.

- Rich config constraint syntax
- Unified type & value constraint
- Configuration conflict checking

Cons.

- Difficult to configuration override for multi-environment scenarios
- Runtime checks and limited performance

Tech.

- CUE
- Nickel
- …

Product

- KubeVela
- …

Modeled KV

Pros.

- Model-centric & constraint-centric
- Scalability on separated block writing with rich merge strategies
- Static type system & analysis
- High Performance

Cons.

- Expansion of different models requires investment in R&D

Tech.

- **KCL**
- …

Product

- KusionStack
- KRM-KCL Tools and Operators