Building infrastructure orchestration software

Patryk Orwat

14.01.2025

How to manage 1M+ resources in a centralized fashion?

Order of battle

- Infra management goal
- K8S concepts
- Hykube playground
- Crossplane case
- Other alternatives
- Lessons learnt

Intro

Why K8S?

- API interface for resource modelling is well defined (better than HATEOAS)
- Developers, operations, testers are familiarized with kubectl
- There’s wide ecosystem around K8S (Helm, Crossplane, Velero, K3S, Argo, …)

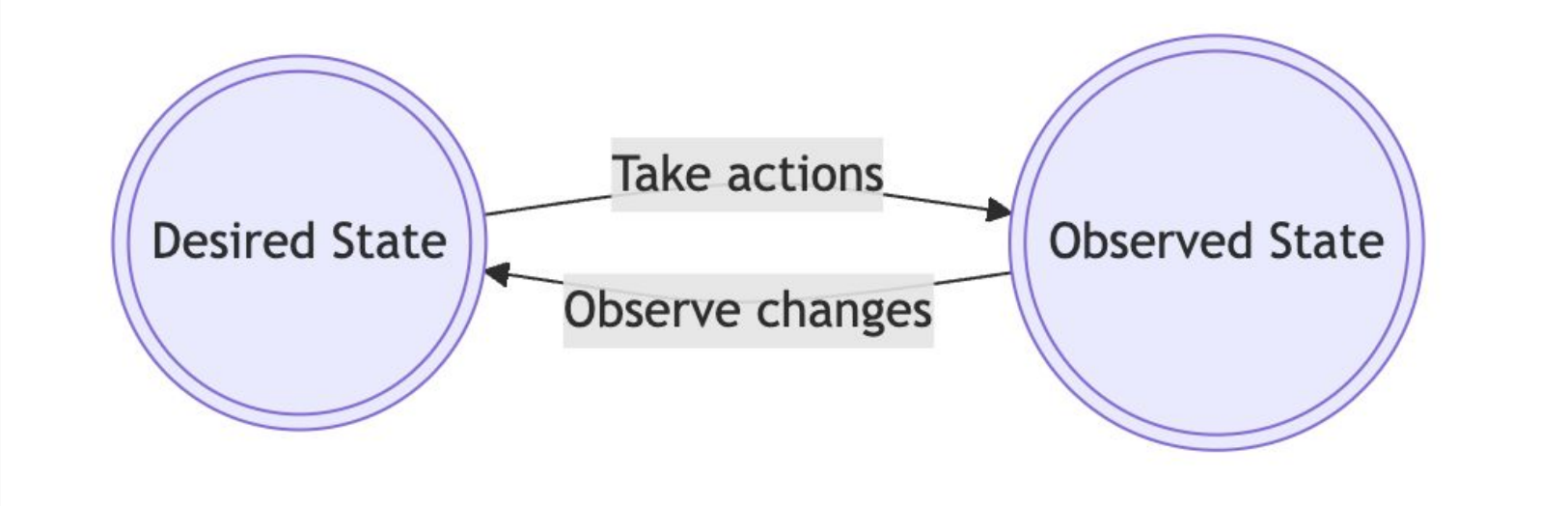
K8S for infra pain points - scalability

- As stated in the documentation for version 1.30.0, it can support
  - up to: 5K nodes
  - 150K pods
  - 300K total containers
- Even though higher numbers are possible, e.g., done by Google, it's still an order of magnitude lower than hyperscale level clusters with 100,000+ nodes.

K8S for infra pain points - containerness bound

- K8S was created to manage containers
- Container-related resources are in the core K8S code
- Reconciliation loop…

K8S for infra pain points - reconciliation loop



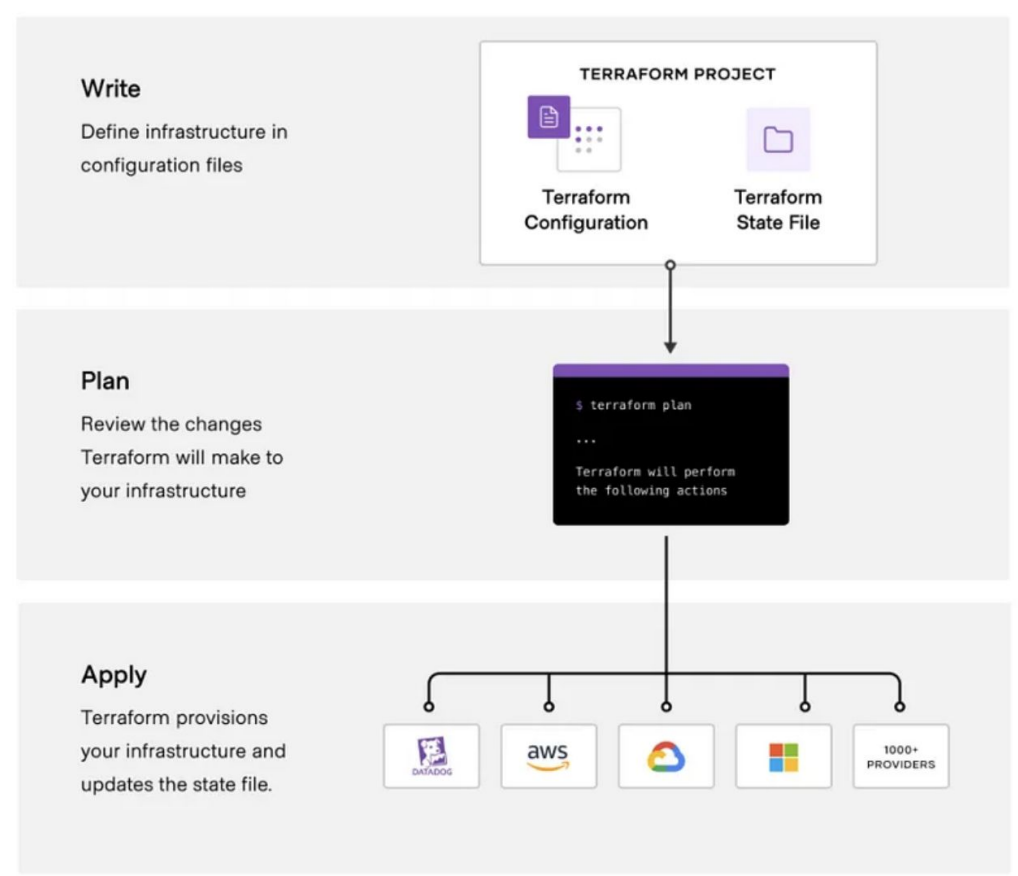
K8S for infra pain points - reconciliation loop

- Loops allow for intent-based management
- Impossible to aggregate reconciliation requests
- Watching mechanism takes up resources

Attempt 1 - hykube

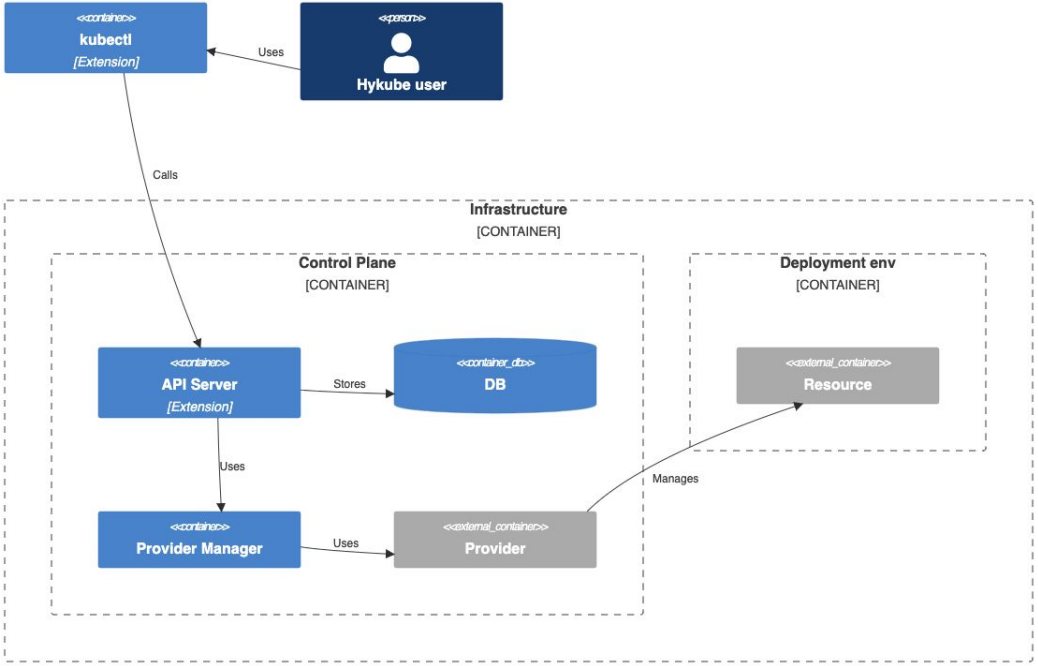
Hykube main points

- Let's walk around reconciliation loop
- Add plan (generate an execution plan) and apply (execute plan actions) commands, see Terraform
- Wrap it with K8S API



Ref: https://medium.com/towards-data-engineering/terraforms-write-plan-apply-but-with-azure-bicep-d09fc3c7d1c3

Hykube Architecture



K8S - kubectl plugins

Adding new commands to plan and apply is easy

Example plugin

```bash
#!/bin/bash

# optional argument handling
if [[ "$1" == "version" ]]
then
    echo "1.0.0"
    exit 0
fi

# optional argument handling
if [[ "$1" == "config" ]]
then
    echo "$KUBECONFIG"
    exit 0
fi

echo "I am a plugin named kubectl-foo"
```

->

To use a plugin, make the plugin executable:

```
sudo chmod +x ./kubectl-foo
```

and place it anywhere in your `PATH`:

```
sudo mv ./kubectl-foo /usr/local/bin
```

->

You may now invoke your plugin as a `kubectl` command:

```
kubectl foo
```

```
I am a plugin named kubectl-foo
```

All args and flags are passed as-is to the executable:

```
kubectl foo version
```

```
1.0.0
```

K8S - API server vs Operator

Custom API server	**Operator pattern (CRDs)**
API server is used (almost) directly by kubectl, via *apiregistration.k8s.io/v1/APIService*	Default K8S API server is used
Flexibility on resource storage	Making use of apiextension API server
Practically, only Go can be used, with forking sample-apiserver	Many languages possible, with Operator Framework/kubebuilder/kopf
You can implement own reconciliation loop	Making use of webhooks and Controllers

Defining resources

K8S resource - definition*

```go
// +genclient
// +k8s:deepcopy-gen:interfaces=k8s.io/apimachinery/pkg/runtime.Object
// +k8s:prerelease-lifecycle-gen:introduced=1.0
// +k8s:prerelease-lifecycle-gen:removed=1.10

type Provider struct {
        metav1.TypeMeta   `json:",inline"`
        metav1.ObjectMeta `json:"metadata,omitempty" protobuf:"bytes,1,opt,name=metadata"`

        Spec     ProviderSpec `json:"spec,omitempty" protobuf:"bytes,2,opt,name=spec"`
        Status   string       `json:"status,omitempty" protobuf:"bytes,3,opt,name=status"`
        Filename string       `json:"filename,omitempty" protobuf:"bytes,4,opt,name=filename"`
}
```

* This is only Go logic; CRDs still require yaml definition (see YAML and Go struct example in Crossplane)

K8S resource - Defining a scheme

```go
var (
	// SchemeBuilder is the scheme builder with scheme init functions to run for this API package
	SchemeBuilder = runtime.NewSchemeBuilder(addKnownTypes)
	// AddToScheme is a common registration function for mapping packaged scoped group & version keys to a scheme
	AddToScheme = SchemeBuilder.AddToScheme
)

// Adds the list of known types to the given scheme.
func addKnownTypes(scheme *runtime.Scheme) error {
	scheme.AddKnownTypes(SchemeGroupVersion,
		&Provider{},
		&ProviderList{},

		&Plan{},
		&PlanList{},
	)
	return nil
}
```

K8S resource - Using a scheme in API server

```go
// New returns a new instance of HykubeServer from the given config.
func (c completedConfig) New() (*HykubeServer, error) {
	genericServer, err := c.GenericConfig.New("hykube", genericapiserver.NewEmptyDelegate())
	if err != nil {
		return nil, err
	}

	s := &HykubeServer{
		GenericAPIServer: genericServer,
	}

	apiGroupInfo := genericapiserver.NewDefaultAPIGroupInfo(hykube.GroupName, Scheme, metav1.ParameterCodec, Codecs)

	v1alpha1storage := map[string]rest.Storage{}
	v1alpha1storage["providers"] = hykuberegistry.RESTInPeace(providerstorage.NewREST(Scheme, c.GenericConfig.RESTOptionsGetter))
	v1alpha1storage["plans"] = hykuberegistry.RESTInPeace(planstorage.NewREST(Scheme, c.GenericConfig.RESTOptionsGetter))
	apiGroupInfo.VersionedResourcesStorageMap["v1alpha1"] = v1alpha1storage

	if err := s.GenericAPIServer.InstallAPIGroup(&apiGroupInfo); err != nil {
		return nil, err
	}

	return s, nil
}
```

K8S resource - Using a scheme in API client

```
crossplaneSchema := runtime.NewScheme()
_ = crossapiextensionsv1.AddToScheme(s)

crossplaneClient, err := client.New(cfg, client.Options{Scheme: crossplaneSchema})
if err != nil {
        return errors.Wrap(err, "cannot create client")
}
```

```
allRevisions := &crossapiextensionsv1.CompositionRevisionList{}
err = c.crossplaneClient.List(ctx, allRevisions, client.InNamespace(ns.GetName()))
```

What I managed to do?

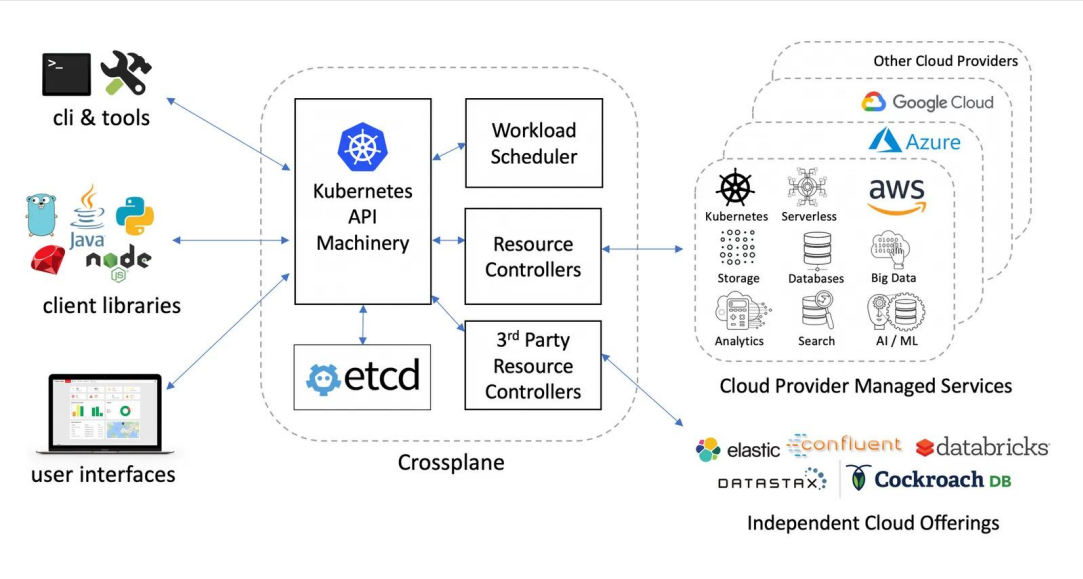
Implemented fetching of a Terraform plugins + creating CRDs

```
(base) patrykorwat@Patryks-MacBook-Pro ~ % kubectl get crd | head -n 10
NAME                                                          CREATED AT
aws-accessanalyzer-analyzers.aws.hykube.io                    2024-09-12T15:06:22Z
aws-accessanalyzer-archive-rules.aws.hykube.io                2024-09-12T15:04:36Z
aws-account-alternate-contacts.aws.hykube.io                  2024-09-12T15:05:28Z
aws-account-primary-contacts.aws.hykube.io                    2024-09-12T15:07:26Z
aws-account-regions.aws.hykube.io                             2024-09-12T15:04:31Z
aws-acm-certificate-validations.aws.hykube.io                 2024-09-12T15:05:22Z
aws-acm-certificates.aws.hykube.io                            2024-09-12T15:05:34Z
aws-acmpca-certificate-authority-certificates.aws.hykube.io   2024-09-12T15:04:08Z
aws-acmpca-certificate-authoritys.aws.hykube.io               2024-09-12T15:06:40Z
(base) patrykorwat@Patryks-MacBook-Pro ~ % kubectl get crd | wc -l
    1372
(base) patrykorwat@Patryks-MacBook-Pro ~ %
```

```
(base) patrykorwat@Patryks-MacBook-Pro ~ % kubectl describe crd aws-s3-bucket | head -n 40
Name:         aws-s3-bucket-accelerate-configurations.aws.hykube.io
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  apiextensions.k8s.io/v1
Kind:         CustomResourceDefinition
Metadata:
  Creation Timestamp:  2024-09-12T15:03:53Z
  Generation:          1
  Resource Version:    52675
  UID:                 4583d2e2-bdc7-4bf5-b511-e014c96e55d3
Spec:
  Conversion:
    Strategy:  None
  Group:       aws.hykube.io
  Names:
    Kind:       aws-s3-bucket-accelerate-configuration
    List Kind:  aws-s3-bucket-accelerate-configurationList
    Plural:     aws-s3-bucket-accelerate-configurations
    Singular:   aws-s3-bucket-accelerate-configuration
  Scope:        Namespaced
  Versions:
    Name:  v1
    Schema:
      openAPIV3Schema:
        Properties:
          Bucket:
            Type:                                        string
            X - Kubernetes - Preserve - Unknown - Fields:  true
          expected_bucket_owner:
            Type:                                        string
            X - Kubernetes - Preserve - Unknown - Fields:  true
          Id:
            Type:                                        string
            X - Kubernetes - Preserve - Unknown - Fields:  true
          Status:
            Type:                                        string
            X - Kubernetes - Preserve - Unknown - Fields:  true
        Required:
          bucket
```

Attempt 2 - Crossplane

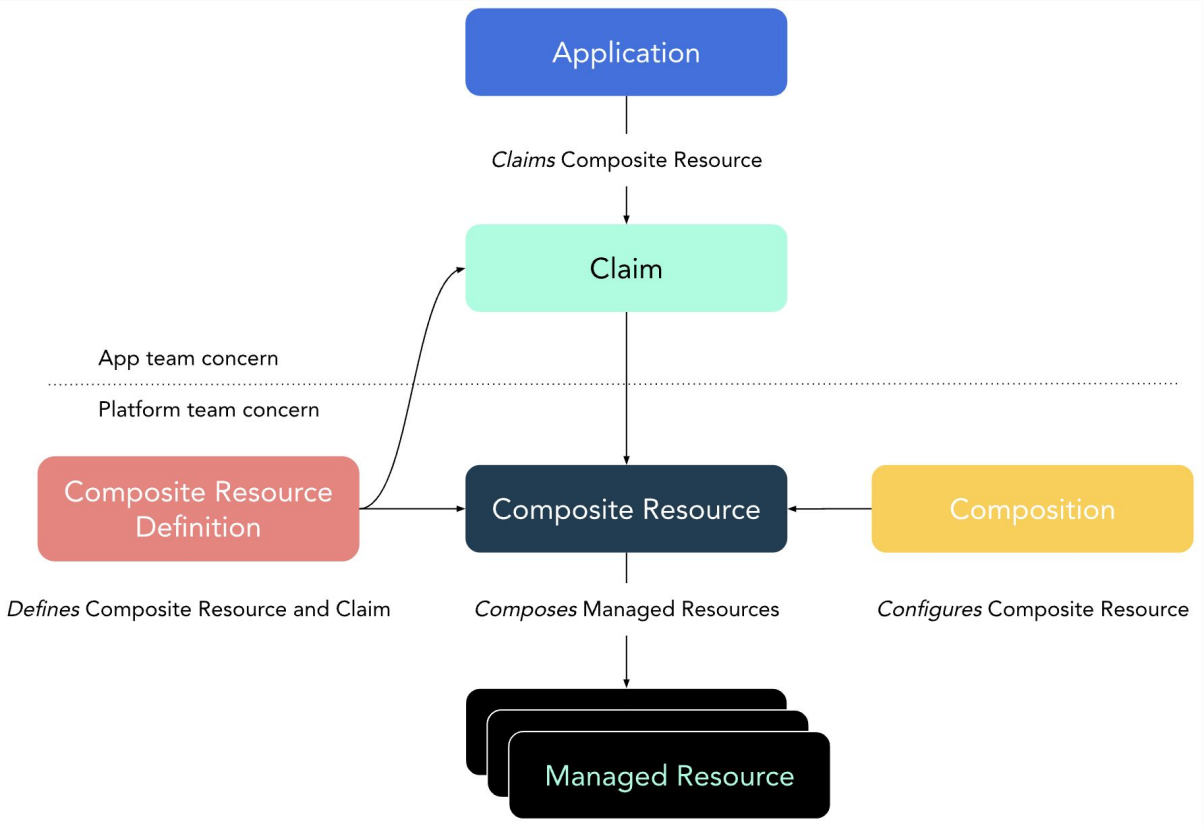
Crossplane - brief overview



Ref: https://blog.upbound.io/introducing-crossplane-open-source-multicloud-control-plane

Crossplane - Composite Resource Definitions

Although https://github.com/crossplane/crossplane/discussions/4178



Ref: https://docs.crossplane.io/latest/concepts/claims/

Crossplane - summary

- Tool to manage external resources with K8S interface
- Adds composite resources
- Has okay-ish support of cloud resources (thanks to Terraform)

Crossplane - “reconciliation” loop

For polled resources (in core AND provider):

- *--poll-interval* - check managed resources for spec changes (default: 1 min)
- *--sync-interval* - check external resources and sync with their state (default: 1 hour)

For watched resources:

- *--enable-realtime-compositions* - turn on XRD watching (turn off pollng)
- –enable-watches - per provider, turn on watching for managed resources

For all resources:

- *--max-reconcile-rate* - rate limiter for REST client & Controller config for MaxConcurrentReconciles; to be configured for core AND per provider

Terraform/Crossplane vs scale

- Core+providers don’t support batch operations
  - Reconciliation loop polls for each resource changes, not in batches
- You must model a batch resource - custom resources
  - Reconciliation loop will pool a batched resource, that will perform optimal-ish requests
  - Max default K8S resource size is 1 MB (ETCD limit)

Crossplane open issues

- No clear path on managing 1K+ of resources, little talk about it
- Creating XRDs to model batching is painful
  - Kro makes XRDs creation simpler but is not a solution https://github.com/awslabs/kro
  - Need a dedicated model - doesn’t seem hard to do

Lessons learnt

Intent-driven orchestration vs workflows

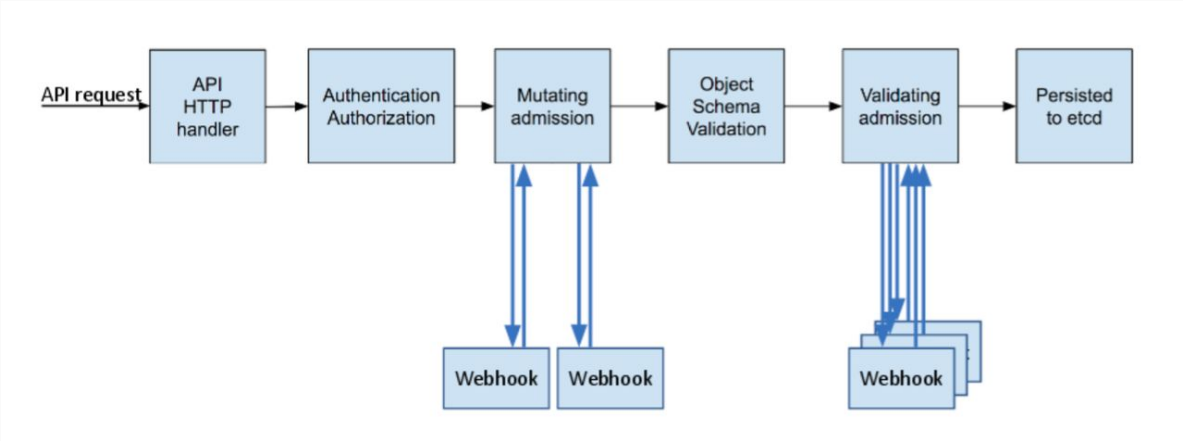
- Intent-driven orchestration:
  - Works well for small quantity of objects (no of batches?)
- Workflows:
  - Define a flow to orchestrate resources

As of today: is there any better Golang OSS than Crossplane + Argo Workflow?

2025 plan - create a resource wrapper + sample workflow

Custom API server vs Operator

- Custom API server works if you:
  - Want to persist data in a separate place
  - Have no reason to make use of reconciliation loop and don't wanna use mutating/validating webhooks



Ref https://kubernetes.io/blog/2019/03/21/a-guide-to-kubernetes-admission-controllers/

External calls always slow things down…

Golang with infra future

Infrastructure from/with Code

		Representation	
		Explicit	Implicit
Domain Language	Expressive	AaC, Modern automation languages	
	Stringly Typed	Traditional IaC	IfC

Ref: https://architectelevator.com/cloud/iac-ifc-trends/

Infrastructure from/with Code vs K8S ecosystem?

- Pulumi - well established project with its own ecosystem
  - Can write custom logic to support non-standard resources? ✅
  - Can write custom workflows? ✅
  - Good base of common cloud resources? ✅
  - Are there standard interfaces? ❌
- Winglang - new language to build infra with code
  - Can write custom logic to support non-standard resources? ✅
  - Can write custom workflows? ✅ (but at what cost?)
  - Good base of common cloud resources? ❌
  - Are there standard interfaces? ❌

OpenAI

openai

Type / to search

Overview | Repositories 183 | Projects | Packages | People 19

Repositories

- All
- Public
- Sources
- Forks
- Archived
- Templates

All

lang:go

7 repositories — Last pushed

openai-go Public
The official Go library for the OpenAI API
Go · Apache License 2.0 · 51 · 835 · 14 · 4 · Updated 3 days ago

kubernetes Public
Production-Grade Container Scheduling and Management
Go · Apache License 2.0 · 40k · 49 · 0 · 0 · Updated on Oct 10, 2024

go-vncdriver Public
Fast VNC driver
Go · MIT License · 62 · 125 · 15 · 2 · Updated on May 20, 2023

oauth2_proxy Public
A reverse proxy that provides authentication with Google, Github or other provider
Go · MIT License · 1.6k · 51 · 0 · 0 · Updated on Apr 2, 2023

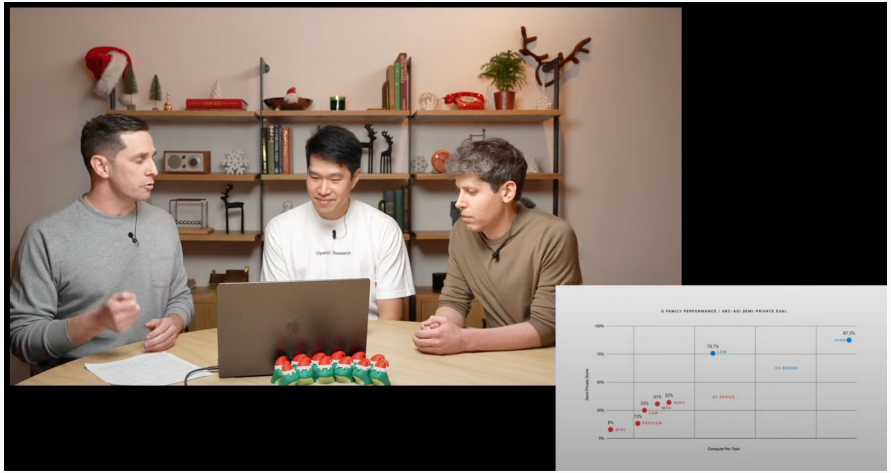
prometheus Public
The Prometheus monitoring system and time series database.
Go · Apache License 2.0 · 9.3k · 33 · 0 · 0 · Updated on Apr 2, 2023

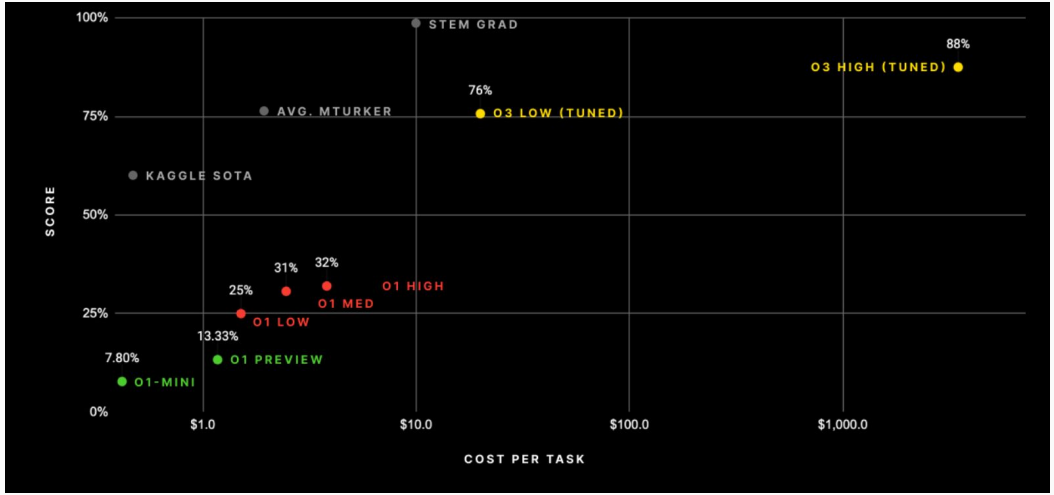
scheduler-plugins Public
Repository for out-of-tree scheduler plugins based on scheduler framework.
Go · Apache License 2.0 · 536 · 11 · 0 · 1 · Updated on Apr 2, 2023

go-retryablehttp Public
Retryable HTTP client in Go
Go · Mozilla Public License 2.0 · 257 · 12 · 0 · 0 · Updated on Apr 2, 2023

OpenAI

Will AI replace SWEs? There's a lot of marketing and optimism in the field.





Refs:
https://www.youtube.com/watch?v=SKBG1sqdvIU
https://techcrunch.com/2024/12/23/openais-o3-suggests-ai-models-are-scaling-in-new-ways-but-so-are-the-costs/

kthxbye!

Questions?