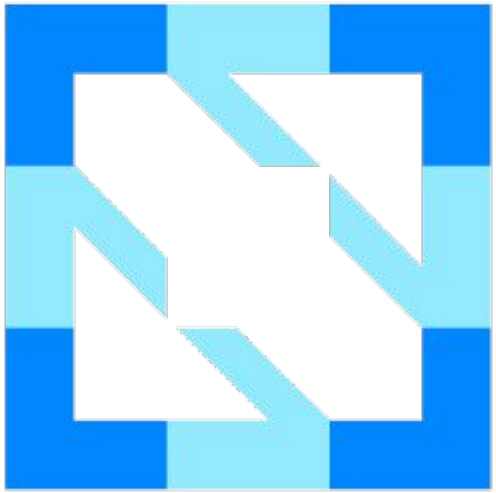
kubernetes

internkurs – del 2

Cloud Native



Hvordan håndtere “YAML-suppe”

kubernetes gir oss standarder for lesbar spesifikasjon i YAML-format

men, hvordan kan vi vedlikeholde denne spesifikasjonen lettest mulig?

verktøy!

“the **package manager** for Kubernetes”

Charts lar oss samle “pakker” med YAML, som enkelt kan installeres og versjoneres av andre

ved hjelp av en fil kalt **values.yaml** kan vi også legge inn justerbare verdier

```
helm install trivy-operator -f values.yaml -n trivy-system
```

Custom Resources og Custom Resource Definitions (CRDs)

lar oss utvide Kubernetes sitt API med “**custom**” ressurser

CRD:

- hvordan skal spec for ny ressurs se ut

CR:

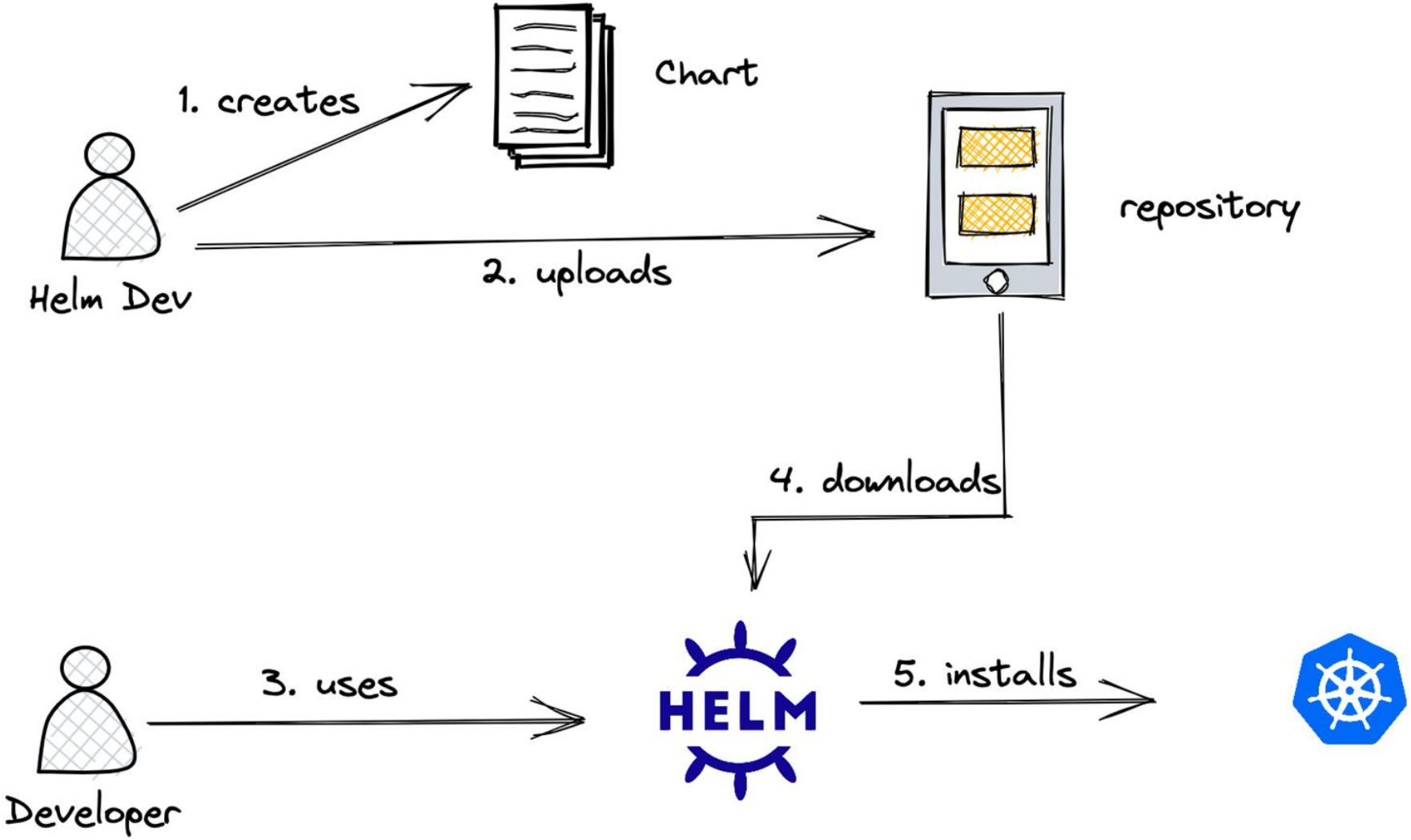
- den faktisk nye ressursen

Custom Resource Definitions (CRDs)

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: crontabs.stable.example.com
spec:
  group: stable.example.com
  versions:
    - name: v1
      served: true
      storage: true
      schema:
        …
  scope: Namespaced
  names:
    plural: crontabs
    kind: CronTab
```

Custom Resource Definitions (CRDs)

```
apiVersion: stable.example.com/v1
kind: CronTab
metadata:
  name: my-new-cron-object
spec:
  …
```


1. creates
Chart
Helm Dev
2. uploads
repository
4. downloads
3. uses
HELM
5. installs
Developer

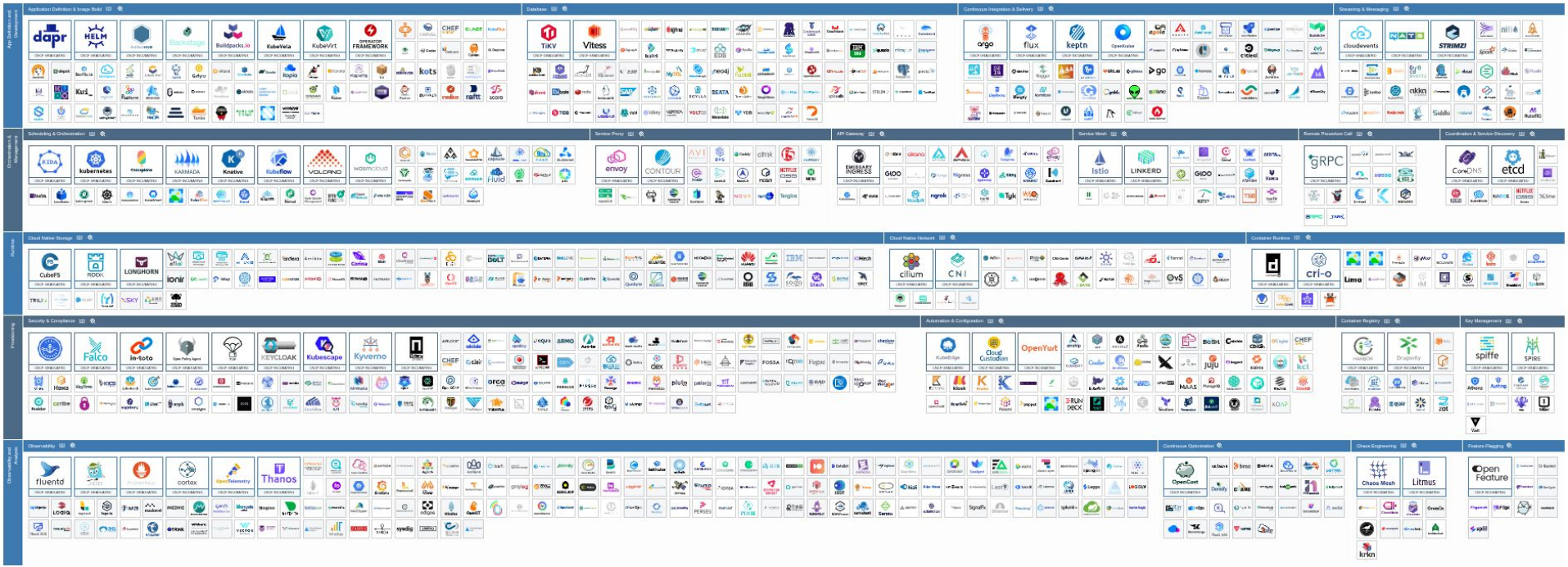
bruker templating for å generere YAML, med verdier fra **values.yaml**:

```
apiVersion: v1
kind: Pod
metadata:
  name: {{ .Values.name }}
  namespace: {{ .Values.namespace }}
spec:
  containers:
    - image: nginx:latest
      name: nginx
```

kan brukes som...

- “maler” for utviklingsteam som skal rulle ut applikasjoner
- måte å **installere applikasjoner fra tredjeparter**

Cloud Native Landscape



Cloud Native + Kubernetes = din helt egen sky!

Trenger du…

Kafka? `helm install strimzi oci://quay.io/strimzi-helm/strimzi-kafka-operator`

Postgres? `helm install cnpg cnpg/cloudnative-pg`

Prometheus? `helm install prometheus prometheus-community/prometheus`

Grafana? `helm install my-release grafana/grafana`

Elastic? `helm install elastic-operator elastic/eck-operator`

Imperativt vs. deklarativt

ved hjelp av Helm kan vi **imperativt** installere ressurser på clusteret vårt:

```
helm install …
```

dette ligner på:

```
kubectl create …
```

…men hvordan kan vi gjøre alt dette **deklarativt**, på en skalerbar måte?

GitOps!

GitOps

istedenfor å “**pushe**” endringer, så “**puller**” vi spesifikasjonen direkte fra et Git-repo

med imperativ “pushing” har vi lite kontroll på når/hvordan/hvorfor ting skjer, og det skalerer dårlig med mange clustere (da må du pushe 100 ganger for 100 clustere)

heller ingen kontroll på hva som skjer etter “**push**”, med GitOps vil vi hele tiden synkronisere konfigurasjon i Git og hva som faktisk er i cluster (**reconcile**)

hvis alle endringer som noen gang har skjedd ligger sentralt i et felles Git-repo har vi **full kontroll**

GitOps

å lage nye clustere blir trivielt: bare pek det nye clusteret på Git-repo, og voila så har det et helt nytt produksjonsklart cluster med alle operators og applikasjoner kjørende

“**cattle not pets**”: bytt ut alt istedenfor å oppgradere

“disaster recovery” blir også trivielt; hvis noe går galt med clusteret og alt blir slettet, er det bare å peke på konfig i Git, og voila er alt oppe igjen

Argo CD

en av de mest populære GitOps-verktøyene

platform-utviklerens beste venn

installeres på clusteret og pekes mot et Git-repo

kan også installeres på et sentralt cluster for å styre andre eksterne clustere (**control plane**)

følger med et **GUI** der du kan se ressurser i cluster(ene)

demo!

spørsmål?

oppgaver

baksetercx/kubernetes-internkurs