API Design Lessons From Kubernetes
https://manifold.co

James Bowes
Technical Lead
KUBERNETES?
THE CLIENT API
kubectl

your client

API server
your client

fetch existing version

API server

examine and make changes

your client

send new version

API server

Does
resourceVersion
field match?
yes: accept
no: reject
PATCHING
JSON Merge Patch

- Plain JSON document only containing changes
- Declarative and simple
- RFC 7386
JSON Merge Patch

... metadata:
  labels:
    service: worker
    app: marketplace
spec:
  containers:
    - image: manifold/worker:1.31.18
      name: worker
metadata:
  labels:
    service: billing-worker
spec:
  containers:
    - image: manifold/sidecar:1.0.0
    name: metrics
metadata:
  labels:
    service: billing-worker
    app: marketplace
spec:
  containers:
    - image: manifold/sidecar:1.0.0
      name: metrics
metadata:
  labels:
    service: billing-worker
    app: marketplace
spec:
  containers:
    - image: manifold/sidecar:1.0.0
      name: metrics
JSON Merge Patch

- Objects are always merged
- Arrays are always replaced
- No explicit null

See also JSON Patch *(RFC 6902)* - Operation based
Strategic Merge Patch

- Adds replace, merge, delete for objects and arrays
- Merge keys for array elements
- Schema defined defaults
- Client directive overrides
Strategic Merge Patch

...

metadata:
  labels:
    service: worker
    app: marketplace

spec:
  containers:
    - image: manifold/worker:1.31.18
      name: worker
Strategic Merge Patch

```yaml
spec:
  containers:
    - image: manifold/sidecar:1.0.0
      name: metrics
```
Strategic Merge Patch

... spec:
  containers:
  - image: manifold/worker:1.31.18
    name: worker
  - image: manifold/sidecar:1.0.0
    name: metrics
Strategic Merge Patch

... spec:
  containers:
    - image: manifold/worker:1.31.18
      name: worker
    - image: manifold/sidecar:1.0.0
      name: metrics
Strategic Merge Patch

```yaml
spec:
  containers:
  - image: manifold/sidecar:1.0.0
    name: metrics
  - $patch: replace
```
LONG RUNNING OPERATIONS
Queue Endpoint

PUT https://my.api/songs/2

HTTP/1.1 202 Accepted
Location: /queue/2890

status: "pending"
lastMessage: "request accepted"
created: "2018-09-24T13:35:05Z"
Queue Endpoint

- Good for infrequent operations
- Requires clients to understand two resource types
- What about resources that are continually experiencing long running changes?
Spec and Status

- Include operation status in the resource
- **spec** describes a resource’s desired state
- **status** describes a resource’s current state
- **status** does not mirror **spec**
apiVersion: extensions/v1beta1
kind: Deployment
spec:
  replicas: 2
...
status:
  availableReplicas: 2
...
THE INTERNAL MODEL
Edge Triggered

Trigger (rise)

Trigger (fall)

Level Triggered

Trigger high

Trigger low

Trigger low
LOST DATA
SLOW REACTIONS
PREVENTION
AND REPAIR
Lost Intermediate Data

● Edge triggered
  ○ Track last seen revision number
  ○ Perform a resync on missing version

● Not an issue for level triggered
Stale Data

- Heartbeat
- Periodic resync (done in Kubernetes)
Slow Reactions

- Edge triggered
  - Keep state for last value
  - Compute change to apply with real value, last seen, and changed
- Not an issue for level triggered
TCP implementations should follow a general principle of robustness: be conservative in what you do, be liberal in what you accept from others.

- Jon Postel, RFC 761
be exhaustive in what you send, be explicit in what you receive from others.
COLOCATE COMPLEX LOGIC WITH STATE
IDENTIFY THE AUDIENCE AND BUILD FOR THEM
MAKE IT HARD FOR THE AUDIENCE TO FAIL
Thank you!