

Data Sovereignty Is Built, Not Bought

Aarno Aukia, Partner, VSHN
Cloud Native Days Austria
29 Sep 2026



Cloud Native Days Austria
2026

Data Sovereignty Is Built, Not Bought

Aarno Aukia, Partner, VSHN
Cloud Native Days Austria
29 Sep 2026

The deal we were offered

THE OFFER

Modern managed cloud services.

Databases, queues, AI: self-service, in minutes.



THE PRICE

Your data moves into our datacenter.

And the jurisdiction follows the provider.

The law follows the provider, not the rack.

Executive order: your service can be switched off.

CLOUD Act: your data can be handed over.

Data has gravity

Regulation

Health, finance and public-sector data must stay in controlled environments.

Latency

Workloads have to stay close to the systems they talk to.

Volume

Large datasets do not move on a project timeline.

Meanwhile, developers expect the cloud: *“a managed database, self-service, now.”*

“We set the bar very high in terms of data. No financial compensation could ever make up for leaked medical information.”

Mohammad Alavi, CTO, Health Info Net (HIN)

The common mistake

“Let’s move it to a sovereign cloud.”

- 1 A migration project
- 2 A duplicated platform
- 3 A new dependency

You changed whom you depend on, not whether you depend.

EU CLOUD TENDER, APRIL 2026

EUR 180M, scored on sovereignty for the first time.

Three winners at SEAL-3: they “cannot be blocked by a non-EU third party”.

Proximus at SEAL-2: EU companies operating Google Cloud-based infrastructure.

The label said sovereign. The score did not.

Sovereignty, stated as three tests

1

Location

Can you move where it runs without changing how developers consume it?

- ✗ Provider-specific Terraform
- ✓ A Kubernetes service claim

2

Operator

Can you replace who runs it without replacing the technology?

- ✗ A hyperscaler's managed database
- ✓ An open-source operator, config in Git

3

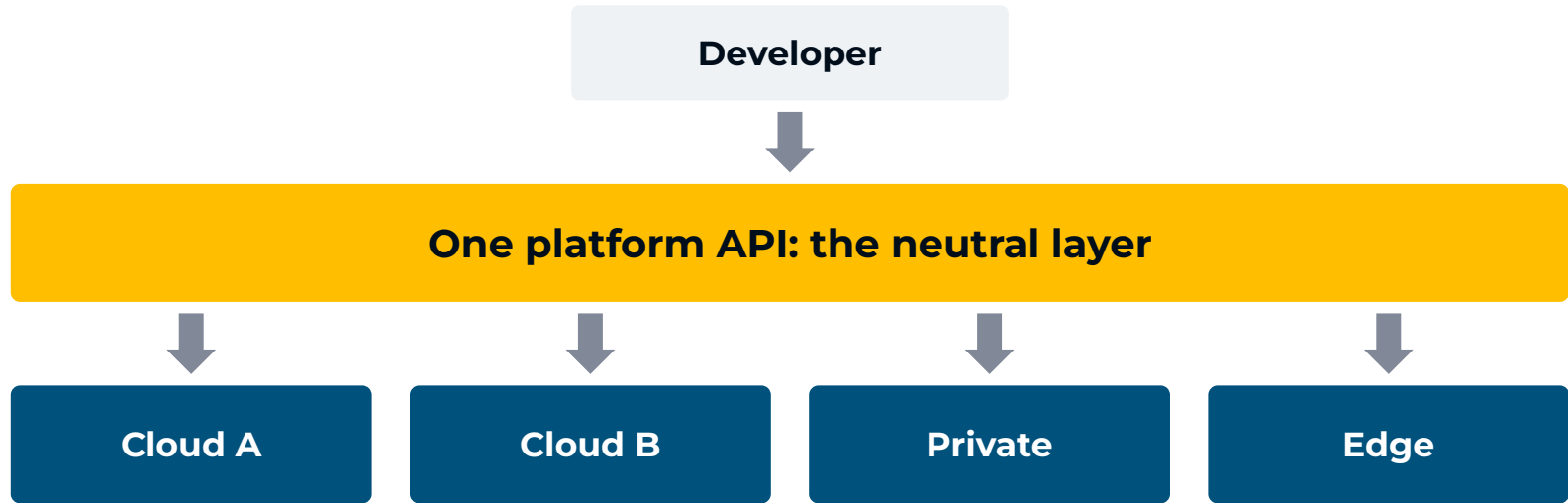
Vendor

Can you replace the software without rebuilding? That one needs open source.

- ✗ VMware after Broadcom
- ✓ Redis → Valkey in eight days

Sovereignty is what you can still replace tomorrow.

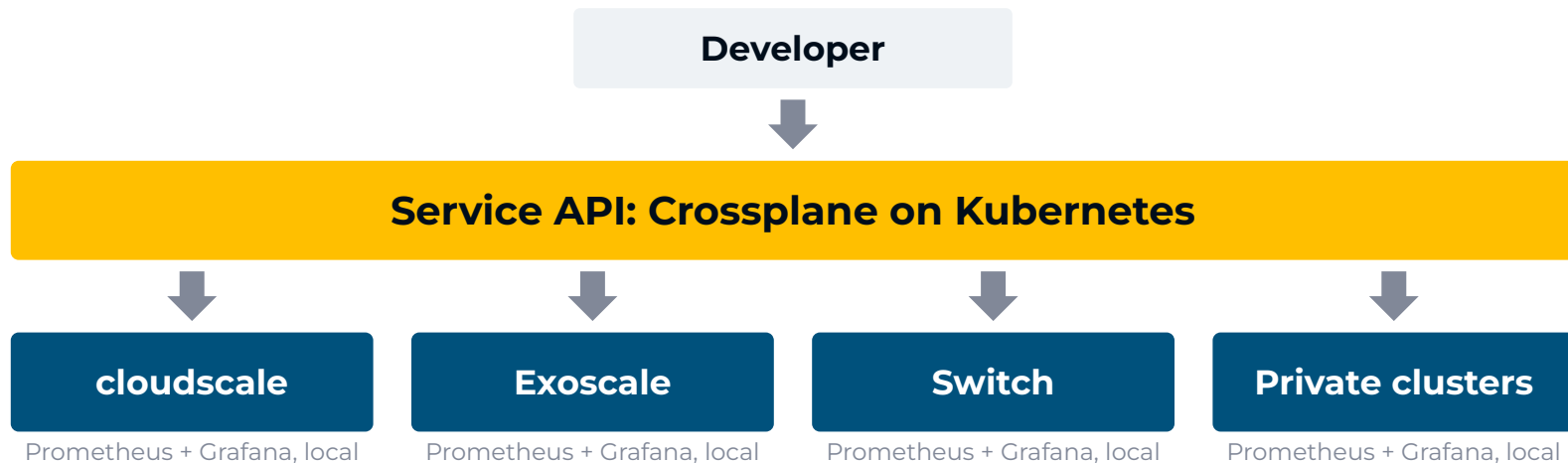
The neutral platform layer



Not avoiding platforms, but designing to survive change.

Case study: VSHN AppCat

AppCat, the VSHN Application Catalog: managed databases and services, ordered as Kubernetes resources, operated 24/7 and delivered where the data is.



Today: PostgreSQL, MariaDB, Redis, Keycloak (with Inventage), Forgejo, Nextcloud, S3 object storage from the underlying infrastructure

Coming: Kafka (with Spoud), OpenBao (with bespinian)

AppCat in production

One service API, two ways to get it. **2,000+ managed instances since 2021.**



Private: dedicated clusters

Every managed OpenShift cluster we run includes AppCat by default.

In the customer's own infrastructure, VMware included, or at a Swiss partner such as cloudscale.

Isolated environments, SLAs up to 99.99%.

Public: shared environments



Order AppCat services like SaaS, at cloudscale, Exoscale or Switch.



Shared OpenShift with AppCat built in.

Who runs on it



Mohammad Alavi

CTO, Health Info Net (HIN)

The secure network for Swiss healthcare:
50,000+ healthcare professionals.



Sebastien Pasche

VP Engineering, Taurus

“We reduced monthly incidents from twelve to zero and improved our SLA from 99% to 100%.”

AppCat customers

finnova. acrevis HRM SYSTEMS

 Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Schweizerisches Bundesarchiv BAR
Archives fédérales suisses AFS
Archivio federale svizzero AFS

HIN 

TAURUS

A production database in 10 lines

```
apiVersion: vshn.appcat.vshn.io/v1
kind: VSHNPostgreSQL
metadata:
  name: pgsql-app1-prod
spec:
  parameters:
    size:
      plan: standard-2
  writeConnectionSecretToRef:
    name: postgres-creds
```

Example from the [AppCat docs](#)

No cloud.

No region.

No operator.

No storage class.

What comes back: PostgreSQL with TLS, daily backups, monitoring and a maintenance window, credentials in postgres-creds

What a production team actually sets

```
apiVersion: vshn.appcat.vshn.io/v1
kind: VSHNPostgreSQL
metadata:
  name: pgsql-appl-prod
  namespace: prod-app
spec:
  parameters:
    service:
      majorVersion: "18"
      serviceLevel: guaranteed
  instances: 3
  size:
    plan: standard-2
  backup:
    schedule: "0 22 * * *"
    retention: 6
  maintenance:
    dayOfWeek: tuesday
    timeOfDay: "22:30:00"
  security:
    deletionProtection: true
  writeConnectionSecretToRef:
    name: postgres-creds
```

serviceLevel

The SLA you are buying

instances: 3

High availability

backup

Your recovery point and retention

maintenance

When upgrades happen: on your schedule

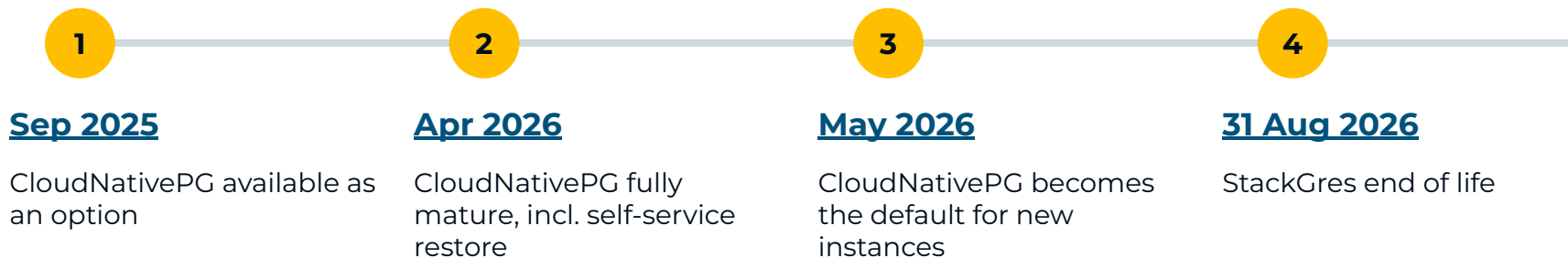
deletionProtection

The data survives a bad kubectl delete

Every block is a decision you own. Still nothing about where it runs.

Same API, new engine

kind: `VSHNPostgreSQL` stayed. The PostgreSQL operator underneath changed.



"Teams already running older PostgreSQL implementations can continue operating them normally, while migration planning can happen on their own timeline."

[VSHN AppCat update, 28 May 2026](#)

We are replacing our own control plane

AppCat on Crossplane
in production 2021 to 2027+



Helmetica

open source, over
time

Why: less development and operations complexity.

One operations framework for any software on Kubernetes, custom or off the shelf.

The framework monitors, backs up and GitOps-manages every service.

Same principle as our Puppet framework on VMs.

**“From Helm chart to
service in 5 minutes.
Every change
transparent.”**

We expect the API to stay
similar, as it did for PostgreSQL.

github.com/helmetica-framework

Before: Redis on Crossplane

```
values := map[string]any{
  "fullnameOverride": "redis",
  "architecture":     "replication",
  "sentinel": map[string]any{
    "enabled": true,
    "masterService": map[string]any{
      "enabled": masterService,
    },
  },
  "replica": map[string]any{
    "replicaCount": comp.GetInstances(),
    "persistence": map[string]any{
      "size": res.Disk,
    },
    "resources": map[string]any{
      "requests": map[string]any{
        "memory": res.ReqMem,
      },
    },
  },
}
```

// ... excerpt

2,076

lines of Go and shell, specific to Redis

11 Go files, 6 shell scripts

to render a Helm chart.

After: Redis on Helmetica

320

**lines: values.yaml, Chart.yaml, 5
small templates**

The upstream chart plus the extras.
Backup, network, maintenance and
credentials come from the framework.

Every change is a diff you can read.

```
# Chart.yaml
dependencies:
- name: azoth # the framework's shared templates
  repository: oci://ghcr.io/helmetica-framework
- alias: redis # the upstream chart
  name: redis-ha
  repository: oci://ghcr.io/helmetica-framework

# values.yaml: every value becomes a field in the API
service:
  '#plan': # ← schema: plan becomes an enum in the API
    enum: [standard-1, standard-2, standard-4, standard-8]
  plan: standard-1
  '#replicas':
    enum: [1, 3]
  replicas: 1
redis:
  replicas: "cel: values.service.replicas"
```

Sovereignty is now measurable

1 Strategic 15%

Who owns and controls the provider?

2 Legal 10%

Which jurisdiction applies?

3 Data & AI 10%

Where is data stored, and who controls the keys?

4 Operational 15%

Can the system be operated independently?

5 Supply chain 20%

How transparent is the supply chain?

6 Technology 15%

Can we migrate away without rebuilding everything?

7 Security 10%

Can security patches be applied independently?

8 Environmental 5%

What environmental impact does the infrastructure have?

Highlighted: decided by your architecture, half the score. EU Cloud Sovereignty Framework, SEAL-0 to SEAL-4; EUR 180M awarded April 2026.

**The question isn't which cloud you choose,
but whether your architecture still gives
you a choice tomorrow.**

Dependent teams ask for permission. Sovereign teams ship.

Thank you and feedback very welcome!



Aarno Aukia

@aarnoaukia a@vs.hn

<https://www.linkedin.com/in/aukia/>



Data Sovereignty Is Built, Not Bought

