What hurts when you use data in operations

— and what I'd do differently

A look at the data plumbing behind RWsOS

DigiShape • Port of Rotterdam • Floris Langeraert • 22-05-2026

RWsOS in one picture

5
operational systems

500
data flows

24/7
early-warning service

3
national met offices

North Sea, IJsselmeer/Markermeer, estuaries, Rhine, Meuse + drought and controlled water bodies.

What does "large" actually mean?

Volume

Terabytes of historical archive, decades of forcing and measurements.

Velocity

Hourly and sub-hourly ingest from dozens of sources.

Variety

Time series, gridded fields, ensembles, point measurements — all mixed.

Reprocessing

If I need to redo this — how long will it take?

Large isn't just TBs. It's also: "if I have to redo this, how long will it take?"

Pain point 1: every vendor is a snowflake

Source	Protocol	Cadence	Surprise
National met offices	REST / files	scheduled	API version cycles
Water boards	various	varies	no shared interface
Foreign authorities	SOAP / FTP	scheduled	silent schema changes
RWS internal	internal feeds	near real-time	ad-hoc additions

The problem

With 5 sources, this is annoying. **With 50+ sources and over 500 data flows, it's existential.**

Every connector is custom. Every change on their side is a surprise on ours.

The problem doesn't scale linearly with sources — it scales with N × rate of change.

Pain point 2: a green pipeline means nothing

- File arrived — but it's empty
- Timestamps in the wrong timezone
- Units silently changed (mm → m, °C → K)
- Pipeline succeeded — forecast was garbage

A green pipeline doesn't mean the data is correct.

Silent failure modes are survivable with 3 pipelines. Invisible with 30.

Scale changes which problems you can survive.

Pain point 3: optimised for one use case

What the source delivers

- **4D ensemble forecast**
  - Hundreds of members × time × space × variables
- **Metadata: provenance, quality, units**
- **Full resolution**

What FEWS receives

- **Time series per location**
  - Ensemble flattened or averaged
- **Metadata: mostly gone**
- **A fraction of the original**

RWsOS does its primary job well. The question is what we'd want for the next generation.

We optimised for early warning. ML, validation, and reprocessing would benefit from richer data upstream.

Three scale walls we're hitting

Storage

Archives grow faster than we can curate them. The fileshare fills up; search time climbs.

Compute

Reprocessing a year of history no longer fits in a maintenance weekend. We can parallelise — but not with this data layout.

Findability

We have the data. Nobody can find it without asking someone who already knows.

Walls we're starting to see, not walls we've crashed into. The system works — the question is the next ten years.

Two sides of the same API

Here's why it matters from the consumer side

Producer

"We expose our data efficiently."

- OGC standard, CoverageJSON
- Spatio-temporal queries, not full downloads
- One source, one API, one audience
- Scale: by design

Consumer

"How do I orchestrate 10 of these at once?"

- Rate limits, quotas, API versions
- Outages at others = outages for us
- Caching, fallbacks, monitoring of their availability
- **But: we see across all suppliers**

The consumer isn't just a victim of API diversity — it's the one who can say what "good" looks like across suppliers.

Where the field is heading

Not fashion — what's left when old tools stop scaling

Storage layer

ZARR, Parquet

Cloud-native formats. Designed for object storage and parallel partial reads.

Query layer

EDR, STAC, OGC APIs

Request slices over the network. No more downloading entire datasets.

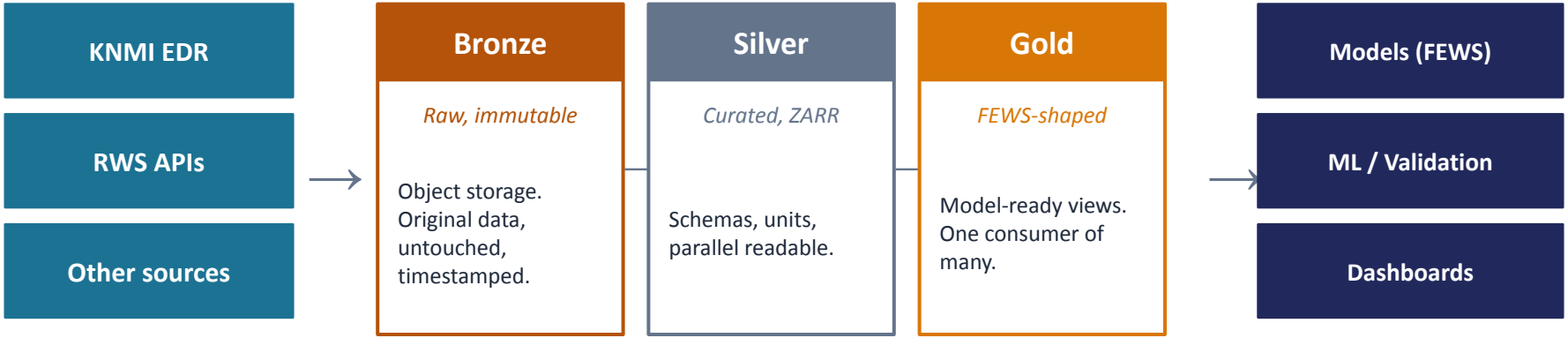
Versioning & catalog

STAC, Earthmover, LakeFS

Git-like versions, snapshots, provenance. Which data went into this forecast run?

Three layers, not three alternatives. Together they change how you think about data pipelines.

What end-to-end could look like



Bronze if we don't keep it, we can never reprocess.

Silver each layer scales on its own terms.

Gold FEWS isn't going anywhere — it's just no longer the only target.

No big-bang migration. Separate the layers so each can scale its own way.

What I'd start tomorrow

Small, measurable, reversible

01

Benchmark one dataset

Convert one curated dataset to ZARR. Compare read times to today's setup. Numbers, not gut feeling.

02

Bronze layer for one shared source

Land one source we all ingest — e.g. cross-border meteo — into object storage. A candidate DigiShape building block, not just ours.

03

Rebuild monitoring properly

Move one pipeline from "script exit 0" to actual data quality checks. Which assumptions are we testing?

Each is doable in a quarter — and the second one only pays off if we do it together.

What we could build together

Exactly what DigiShape exists for: reusable building blocks, shared across the sector

START HERE

Converge on standards

Agree how data is exposed — EDR, STAC, ZARR. Meet the producers halfway. Could start this year.

NEXT

A shared catalog

Federated index across orgs. Find what exists in the sector without knowing who holds it. Nobody gives up control.

AMBITIOUS

A sector reference commons

Canonical, versioned bronze layer for data nobody owns alone — cross-border forcing, boundary conditions.

The alternative: 500 flows × every org, solved separately, with no shared truth. None of us is big enough alone.