Geospatial Sensor Networks and Partitioning Data

SEPTEMBER 2020 | Alex Miłowski, Data Platforms Researcher @ Redis Labs

Introduction

A brief introduction to my background and recent endeavours

Partitioning Sensor Networks

A brief walk through some sensor networks, and geospatial data processing

Evolving to K8s

Collecting, storing, and using geospatial data on K8s

AQI Demo

PurpleAir

A demonstration of computing AQI interpolations

Introduction

redislabs
HOME OF REDIS

Who am I?

My background:

- W3C Invited Expert - various XML, RDFa, Semantic Web technologies, since 1998
- Informatics PhD @ University of Edinburgh - 2014 - Enabling scientific data on the web
- Industry - Variety of startups - large and small.
- Academic - UC Berkeley and SF State - variety of informatics, computer science, and mathematics topics

K8s:

- started in using 2017 - telecom (Orange)
- on-premise and hybrid-cloud for ML “data devops”
- additional focus on data governance vs governance of data

Partitioning Sensor Networks

An exemplar - weather

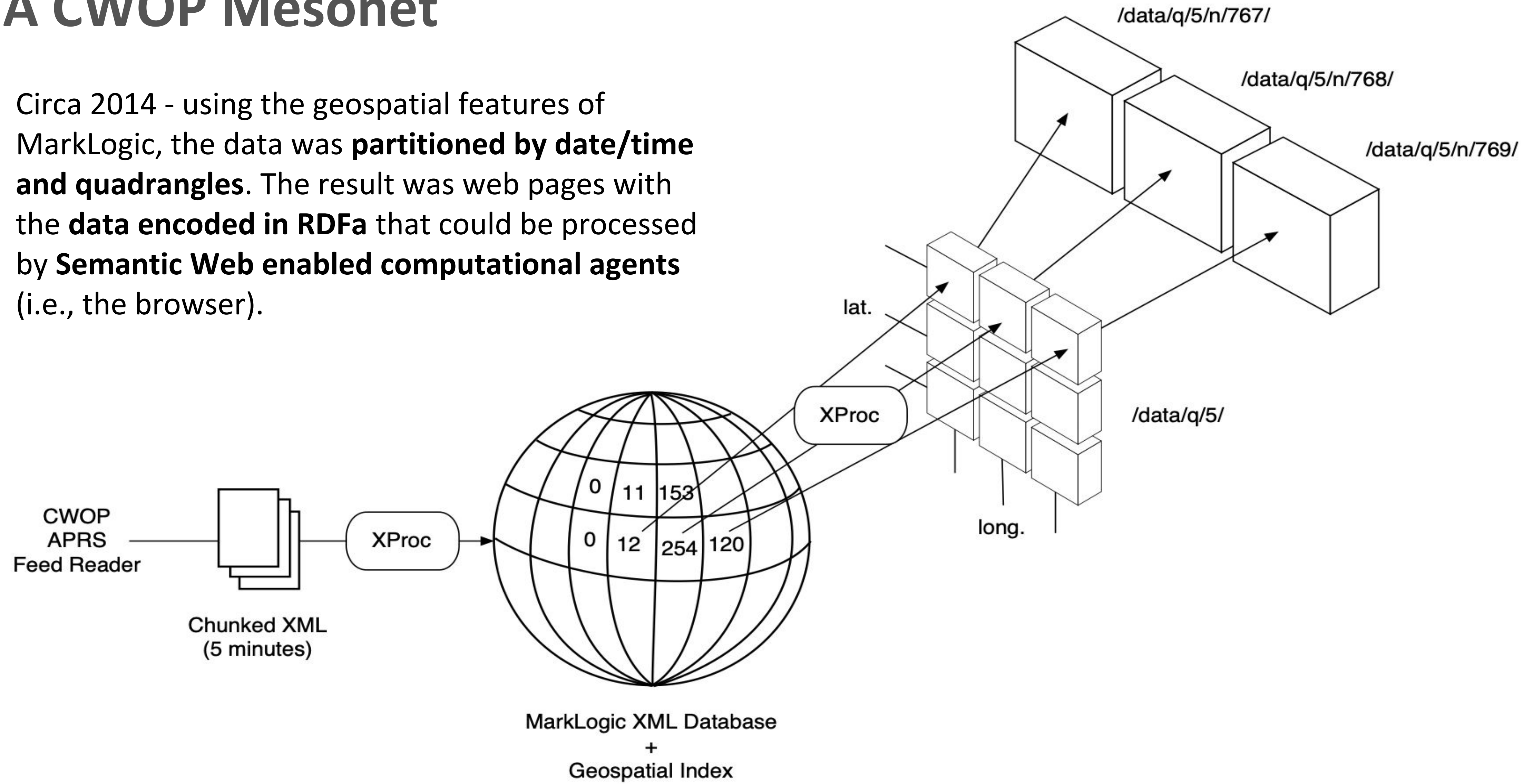
Citizen Weather Observation Program (CWOP)

- a global network of network-connected weather stations
- often run by individuals, businesses, and local government agencies
- a global aggregated data feed over the APRS-IS protocol
- 800 different organizations use CWOP data including the NOAA / NWS in the USA
- ~ 100 million weather reports per month (2014)
- see: http://wxqa.com

DW3904>APRS,TCPXX*,qAX,CWOP:@090158z5132.18N/00043.53W_061/000g001t030r000p000P000h87b10389L000.DsVP
CW1604>APRS,TCPXX*,qAX,CWOP:@090158z4444.70N/06531.17W_204/004g009t027r000p000P000h80b10204.DsVP
DW6741>APRS,TCPXX*,qAX,CWOP:@090158z3749.55N/08000.08W_296/005g...t036r...p...P008h74b10188.DsVP
DW6916>APRS,TCPXX*,qAX,CWOP:@090158z4310.23N/10818.40W_238/001g002t027r000p000P000h58b10189.DsVP
DW6011>APRS,TCPXX*,qAX,CWOP:@090158z4307.07N/08756.60W_261/002g006t028r000p000P000h55b10249.DsVP

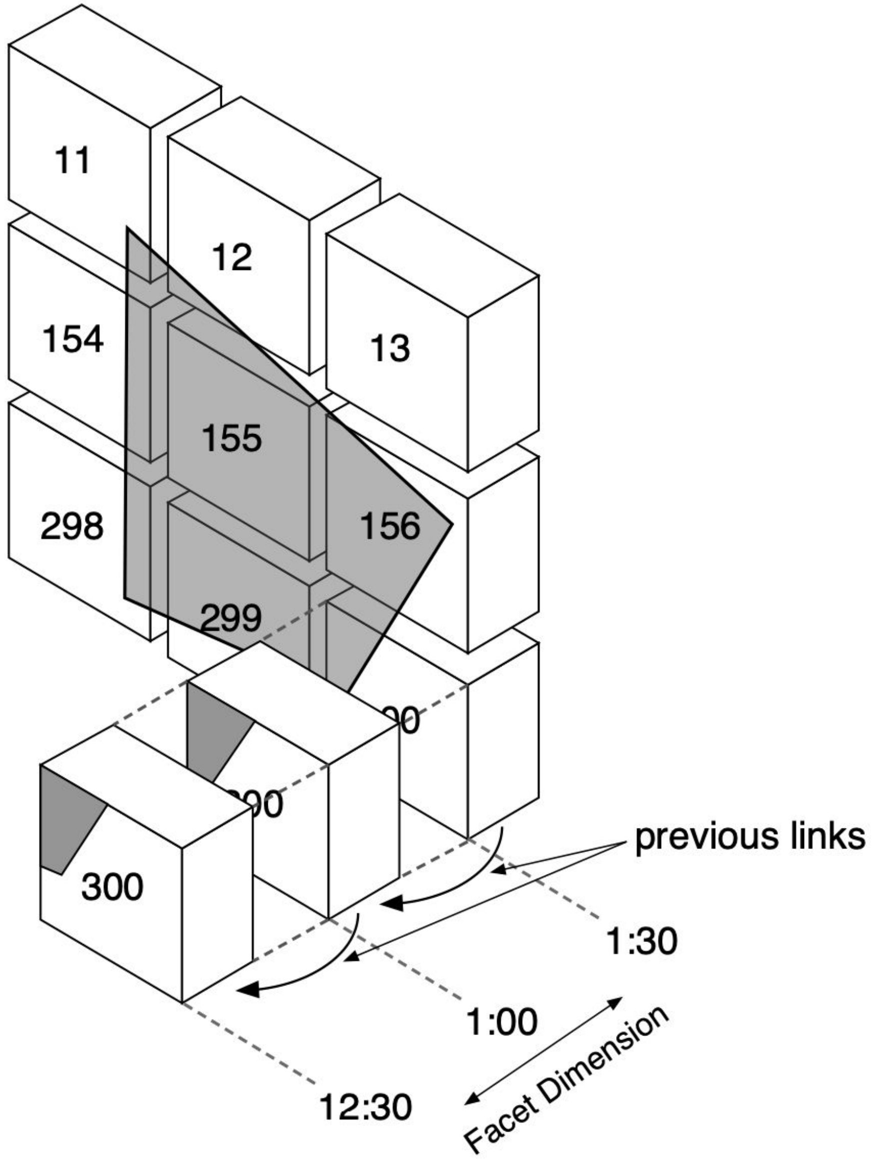
A CWOP Mesonet

Circa 2014 - using the geospatial features of MarkLogic, the data was **partitioned by date/time and quadrangles**. The result was web pages with the **data encoded in RDFa** that could be processed by **Semantic Web enabled computational agents** (i.e., the browser).



PAN Methodology - Key Results

- Partition Annotate and Name (PAN):
  - regular partitions by data facets,
  - enable algorithmic navigation,
  - via annotations and naming,
  - of reasonable-sized data partitions.
- Computations can rely upon consistent data access patterns and latency.
- Naming and annotation give algorithms the essential metadata to navigate the data partitions.
- Standardized formats enable open computation



Barnes Interpolation

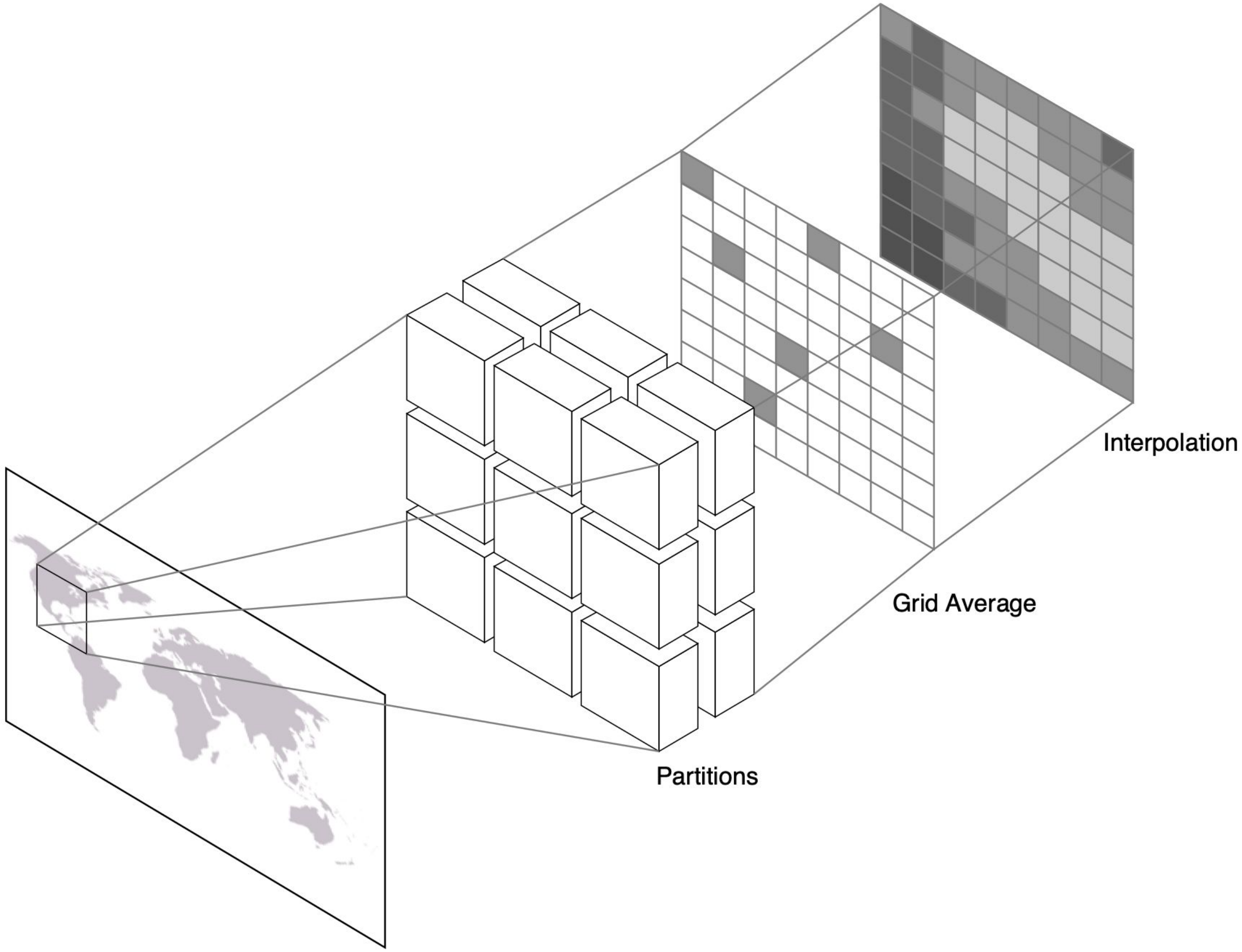
Proposed by Stanley L. Barnes in 1964:

- a method for interpolating a grid of values from a uneven and sparsely populated grid
- dynamic programming algorithm
- feasible with a small amount of memory

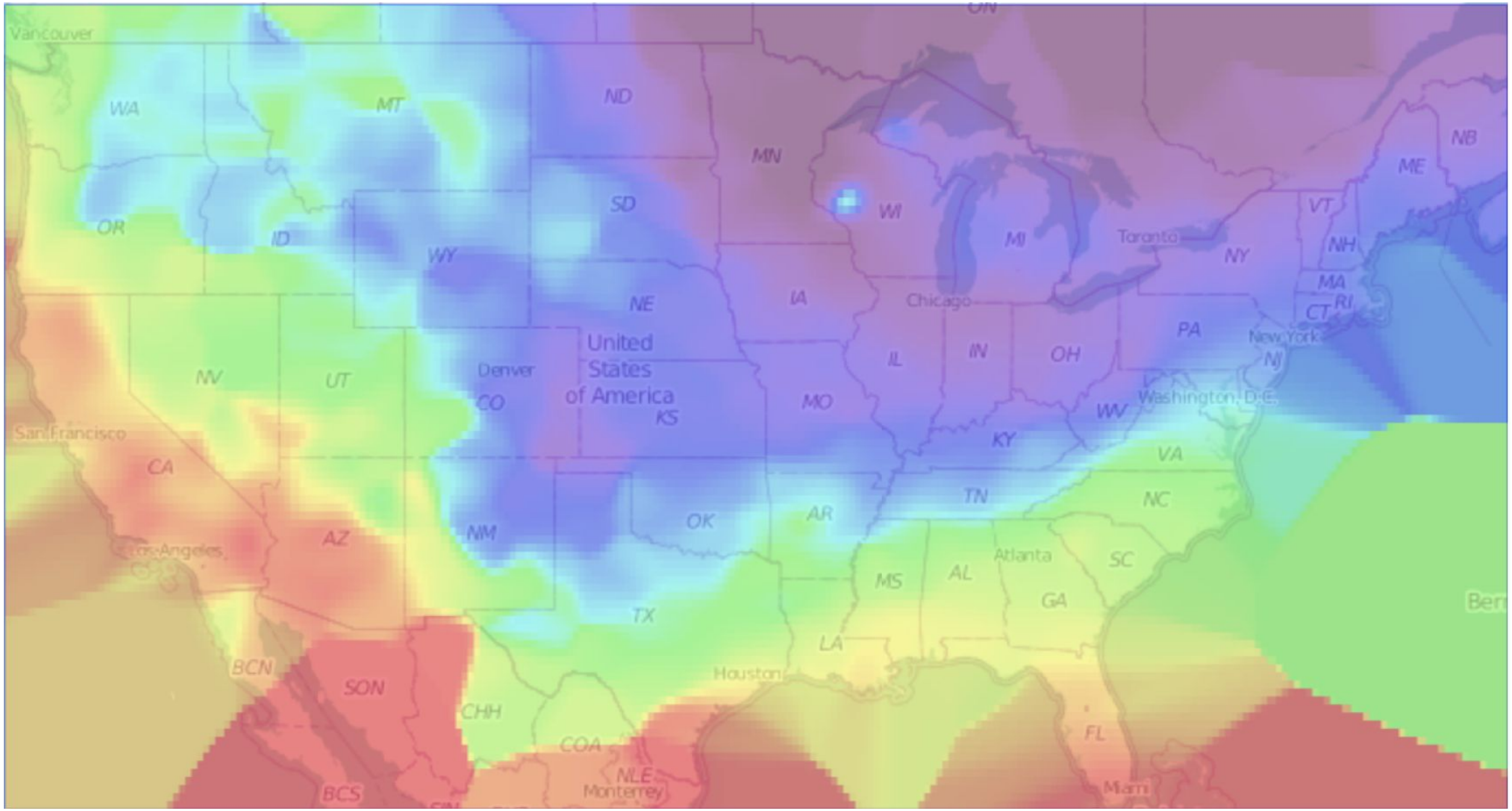
Important for atmospheric interpolation.

Data pipeline is like map/reduce

Think: weather maps!



Polar Vortex - January 23rd, 2014 - PT30M



Real-time computation in the browser - map/reduce - 22 seconds; 13.6 data access ~ 2014

Browsers are a lot faster now!

Challenges - Data DevOps in 2014

- Keeping in collection working:
  - reliability of APRS-IS services
  - reliability of collection service (e.g., restarting on errors)
  - algorithms for retrying service connections via APRS protocol
  - disk space!
- Database sizing / infrastructure sizing
- Keeping ingest working:
  - what to ingest
  - idempotency
  - performance
  - reliability of ingest service (e.g., what to do when things go wrong)

OpenNEX - evolving via Hadoop

NASA OpenNEX Challenge:

- [NASA Earth Exchange]
- a global climate model in HDF5 format
- AWS host data files via S3

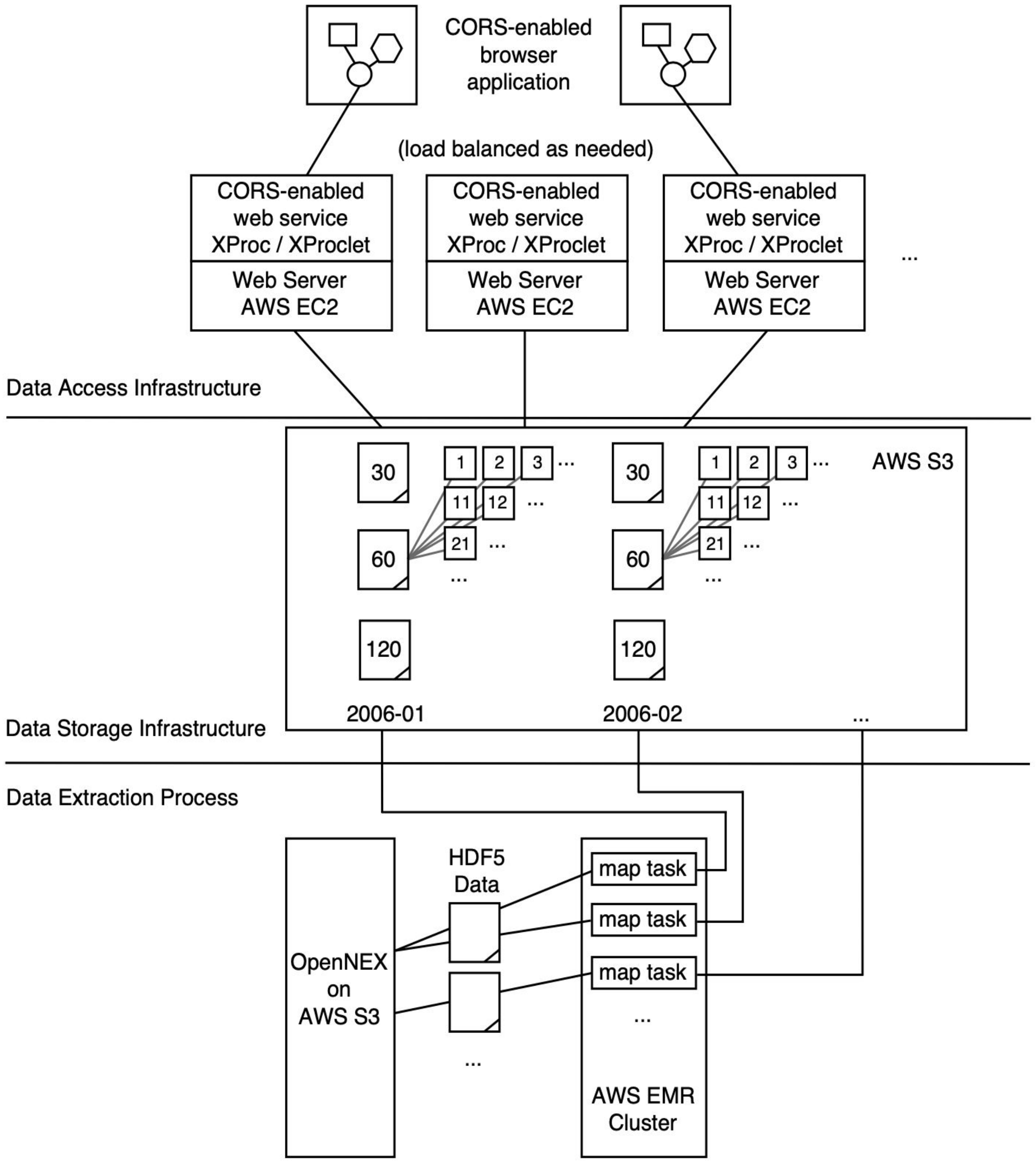
Challenge:

- HDF5 is a challenging binary format
- relatively large file size

Outcome:

- startup an on-demand AWS EMR
- partition into web-accessible data partitions
- processable by browser clients
- my team won an initial award (woohoo!)

See: https://github.com/alexmilowski/opennex



redislabs HOME OF REDIS

Evolving to K8s

redislabs
HOME OF REDIS

Collision course with K8s

Collection: collect and store sensor data as fast and reliable as possible

- APRS/CWOP data source reliability
- long-running jobs with reconnection, servers changes, outages, etc.
- once data is transmitted it is gone if there is no receiver
- downtime of collectors mean coverage gaps

Ingest:

- Data transformation from raw measurements / format
- into cleaned observations (e.g., is this a valid temperature for the atmosphere?)
- in near real-time

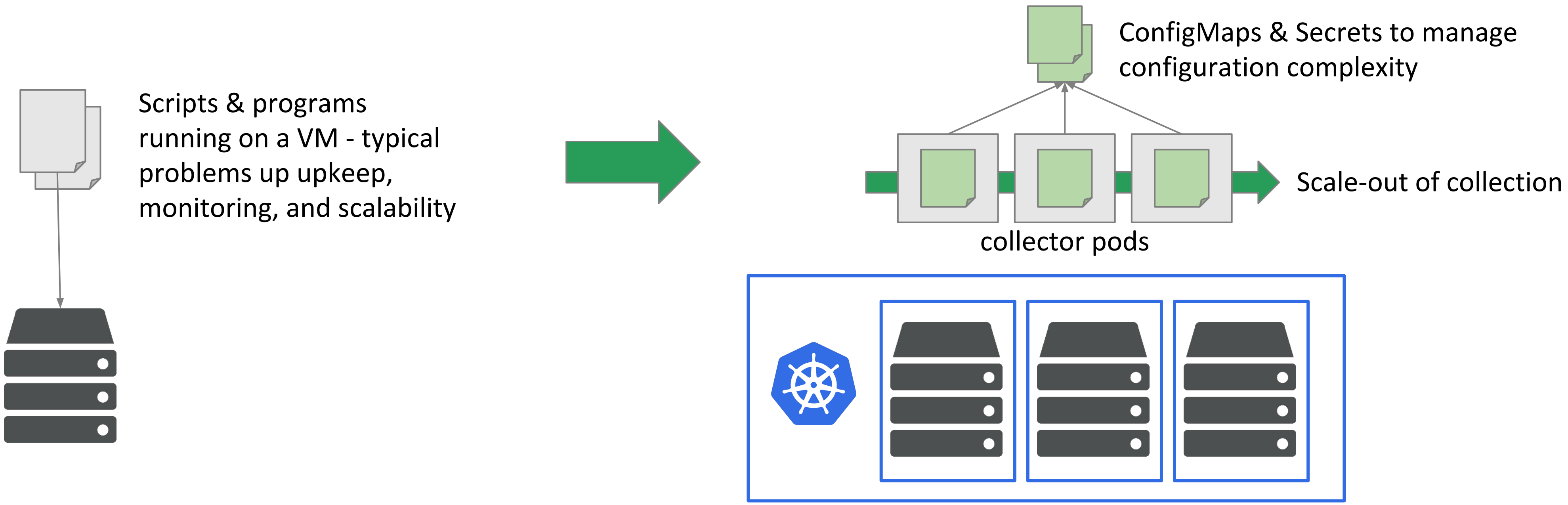
Analysis:

- batch, ad-hoc, or real-time
- application of computations / models to data partitions

Data Collection - clear win for K8s

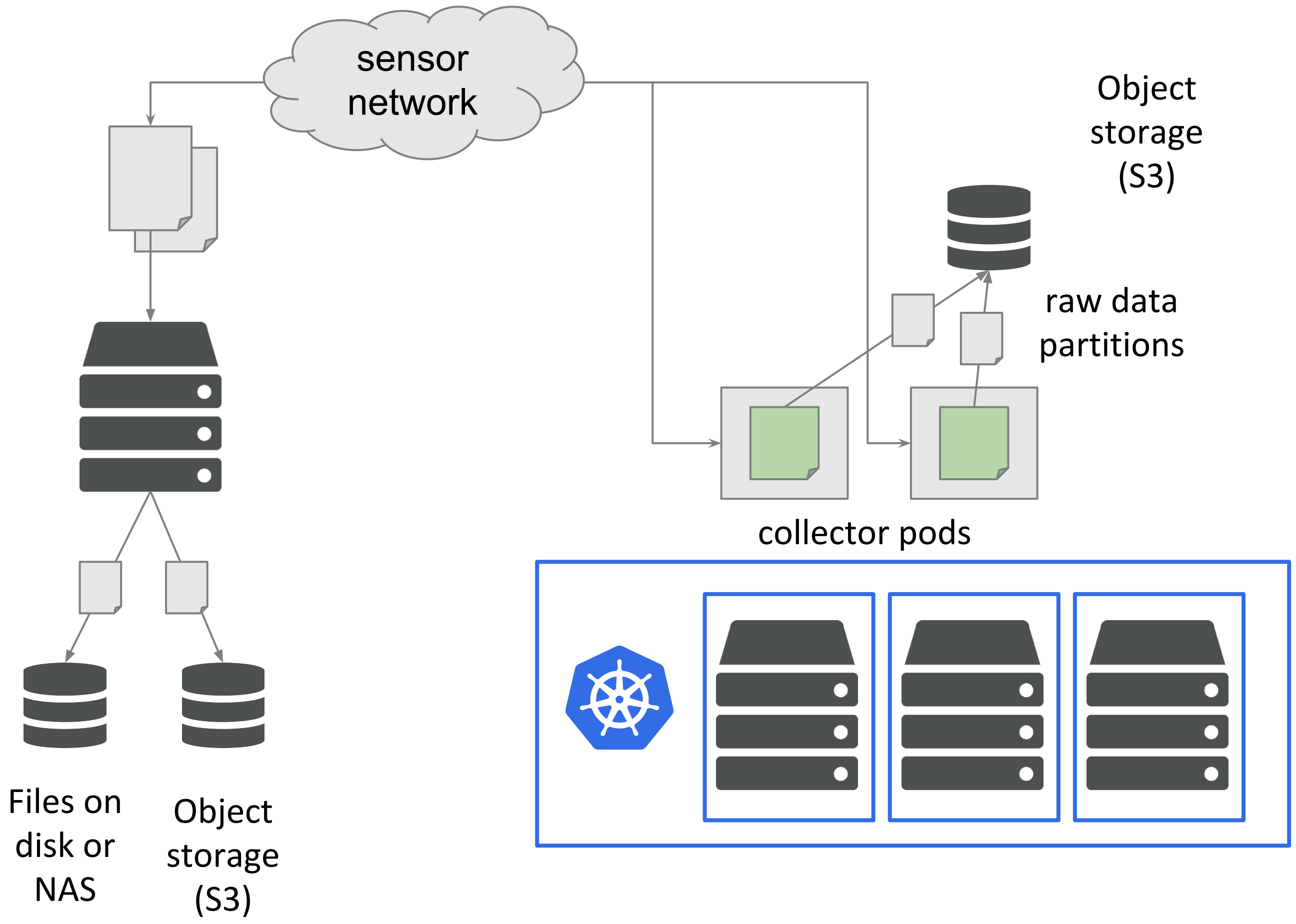
Data collection processes are a clear win:

- long-running processes
- parallelization
- restarting after failures
- operator paradigm provides an evolution path



Where does the raw data go?

Sensor measures are ephemerol.
A principle foundational of data science is to capture as much of the raw data as possible.



✓ Collection partitions data on facets:

- time (~ 5 minute segments)
- source (e.g., server)
- geospatial region

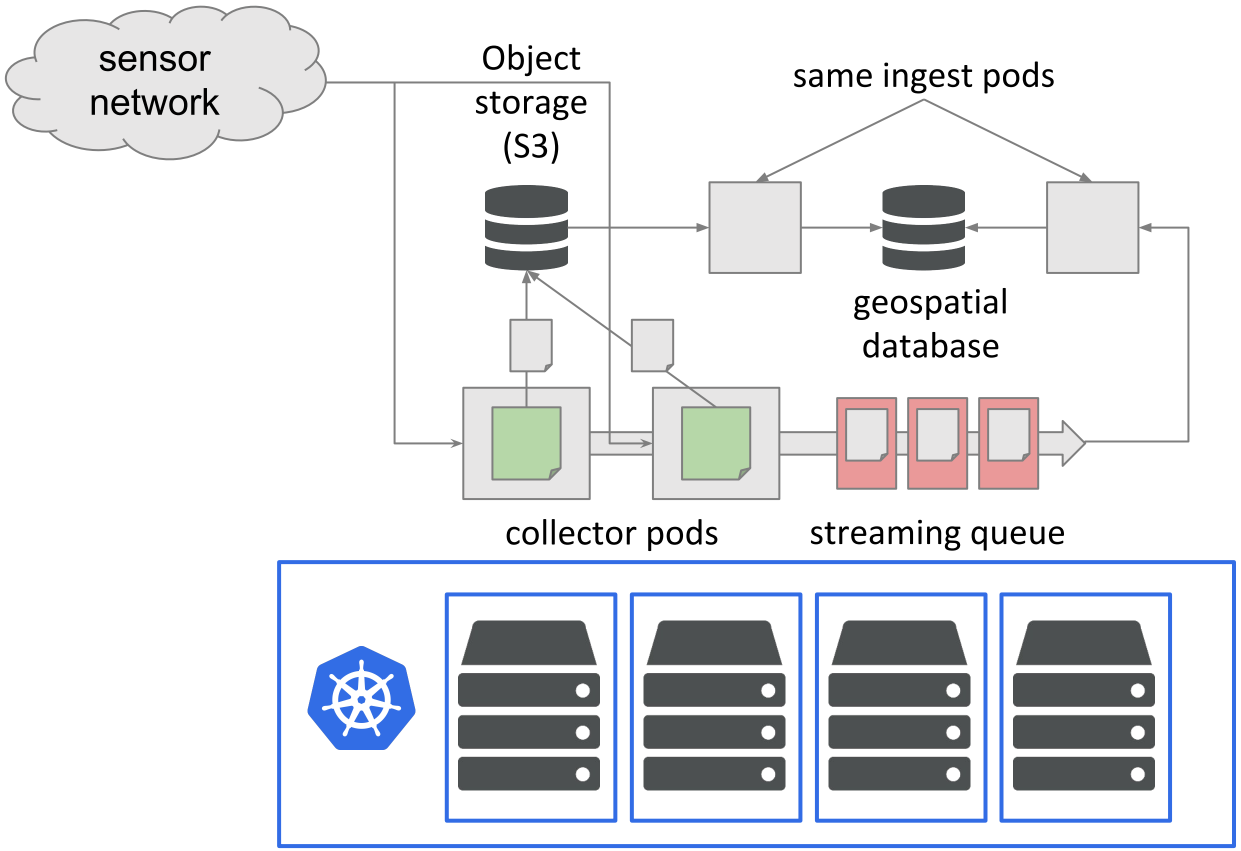
✓ Object storage is still a viable option

✓ Great for commiting ephemerol sensor data.

✗ No real-time ingest

Batch vs "real-time" ingest

Idempotency is a key principle.



✓ ingest via the same process

✓ queues have smaller partitions

✓ you still want a record in object storage

✓ you must be able re-run ingest and get the same result (idempotency)

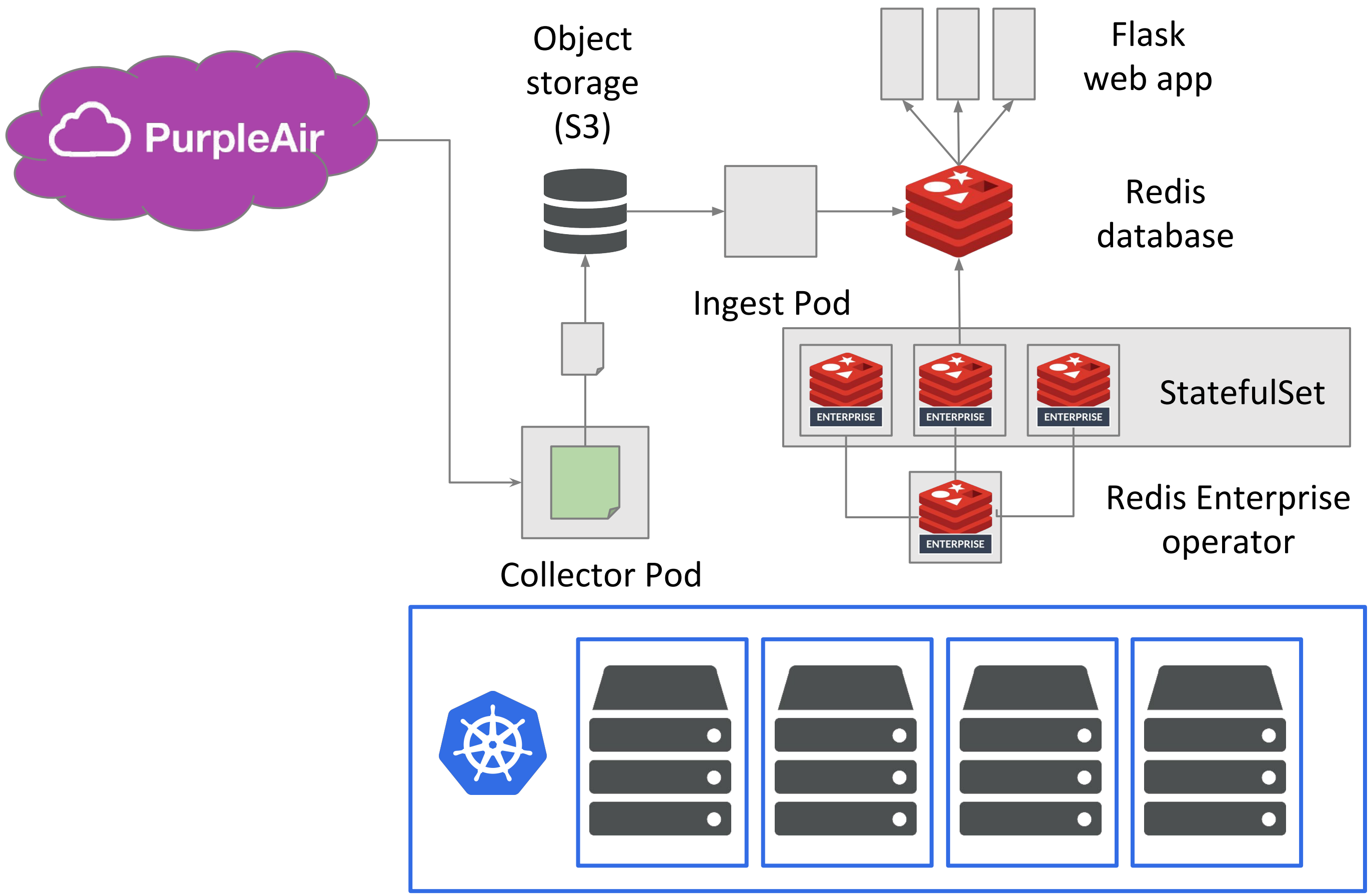
✕ ingestion is not really real-time - partitioning is by the use case (e.g., you don't need real-time air quality; need in 10-30 minute partitions)

AQI Demo

PurpleAir

redislabs
HOME OF REDIS

AQI Interpolation via PurpleAir data



- ✓ single resilient collector
- ✓ parallelizable ingest
- ✓ redis deployment agnostic
- ✓ deploying Redis via operator
- ✓ simple Flask-based web application for visualization of AQI interpolation

Demo time!

Interpolation of Air Quality Index (AQI) via variety of algorithms:

- greater Bay Area AQI
- computed from raw PM measurements from PurpleAir
- collected starting 2020-08-25
- stored in redis using geospatial sets
  - key partition by time period (PT30M)
  - key value: sensor@offset + observations
  - reverse model - key is value, score is geoposition
- simple interpolation options - no AQI model 😞

Controlling Ingest

Ingest jobs are easily scheduled by generating batch jobs (or cron jobs):

```
python job.py --index 1 \
    --type at 2020-09-14T00:00:00,2020-09-14T23:30:00 \
    --name ingest-2020-09-14 | kubectl apply -f -

python job.py --index 1 \
    --type now \
    --name ingest | kubectl apply -f -
```

A template for the job is modified a serialized for submission.

Enhancement: write an operator for managing these jobs for successful completion.