



How DWD downloads and consumes data from WIS2

Global Services

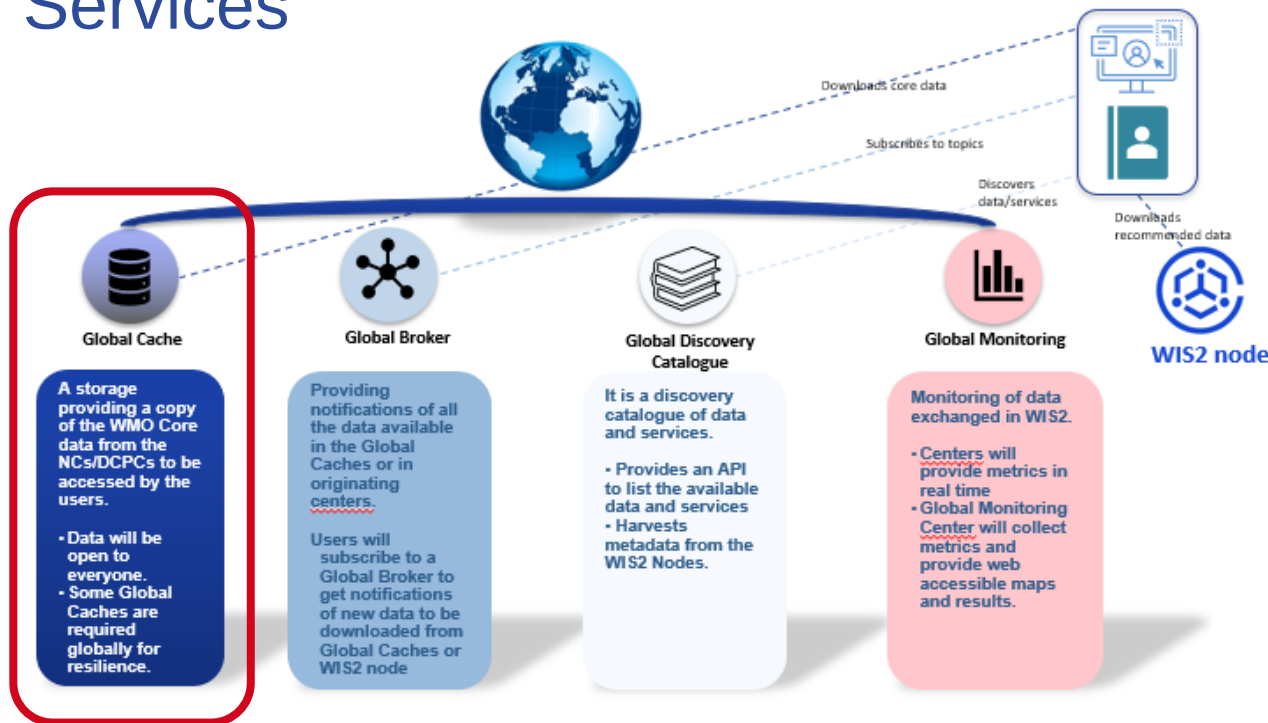


Image: Remy Giraud, MeteoFrance

✓ WHAT IT DOES

- ➔ Mirrors WMO core data (free & unrestricted access) close to consumers
- ➔ Downloads data as soon as a Node's WNM is published, via the canonical link
- ➔ Re-announces availability under its own cache/a/wis2/<centre-id>/... topic
- ➔ Serves that copy from its own storage endpoint — a redundant source if the origin Node is slow or unreachable
- ➔ takes the load off the node
- ➔ Connects to at least two Global Brokers for redundant ingestion
- ➔ Exposes monitoring metrics to the Global Monitor

✗ WHAT IT DOESN'T DO

- ➔ Cache recommended (access-controlled) data — that stays with the origin Node only
- ➔ Replace the Node as the authoritative source — the origin copy is still canonical
- ➔ Store data indefinitely — cached copies are cleaned up after a short retention window (24h, not months)
- ➔ Handle discovery metadata — that's the Global Discovery Catalogue's job
- ➔ Guarantee it holds every core dataset at every instant — it's best-effort, near-real-time mirroring

The Caching Flow, Step by Step

Node publishes core data + WNM under `origin/a/wis2/<centre-id>/data/core/...`

Global Cache (subscribed to ≥ 2 Global Brokers) receives the notification

Redundant ingestion — not reliant on a single broker

Downloads the data via the canonical link in the WNM

Stores a copy in its own backend storage

Only core data — recommended data is never mirrored

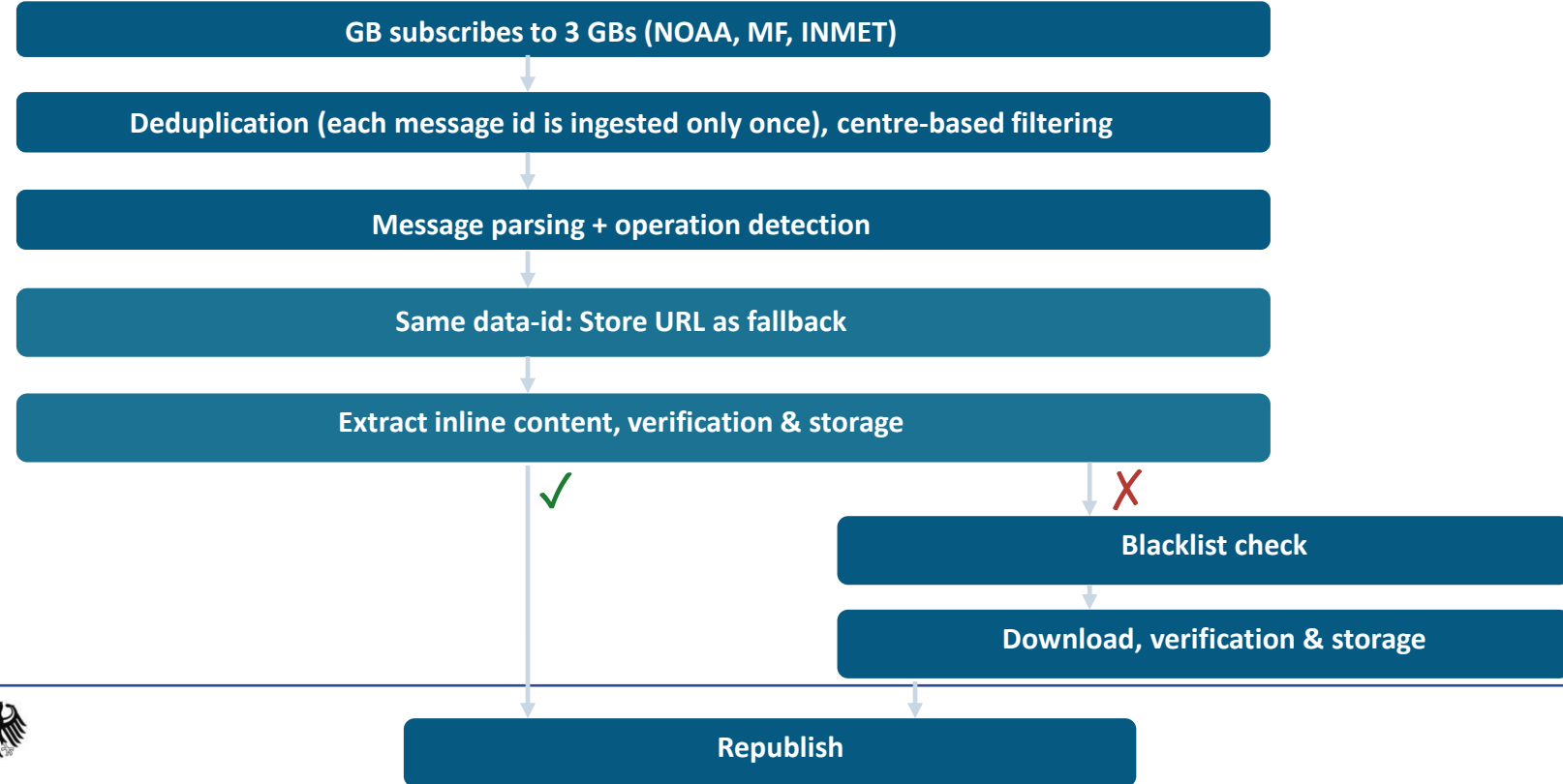
Publishes a new notification under `cache/a/wis2/<centre-id>/data/core/...`

Consumers can now fetch from either `origin/...` or `cache/...`

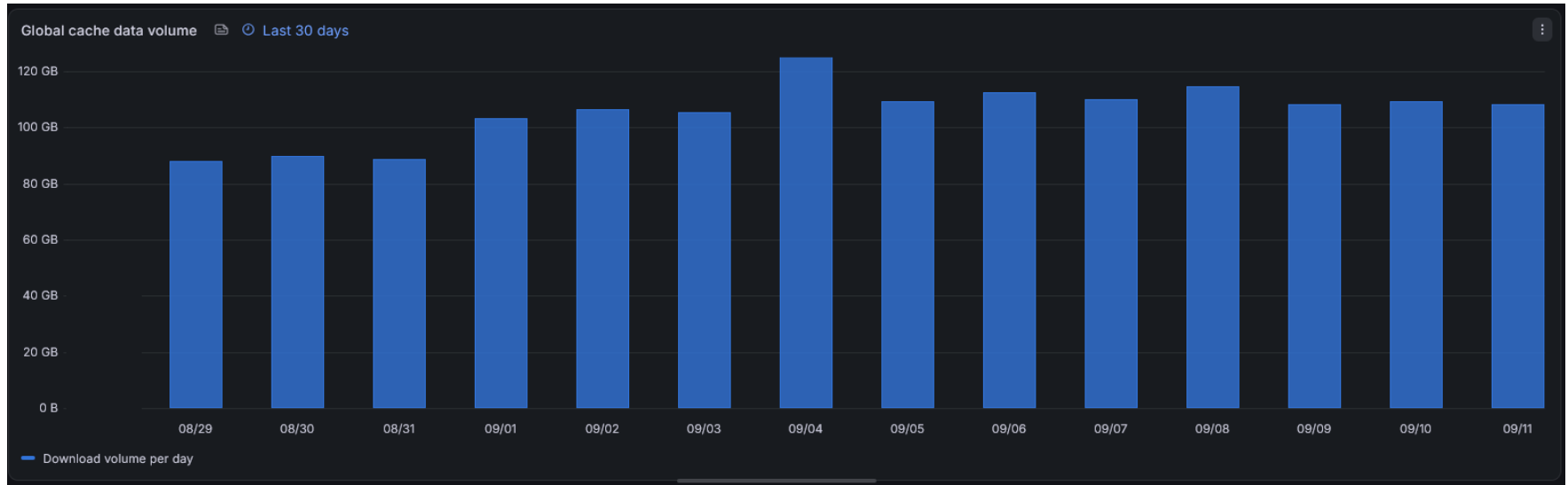
Its recommended to consume from cache

Cached copy is cleaned up after the retention window (default ~24 h)

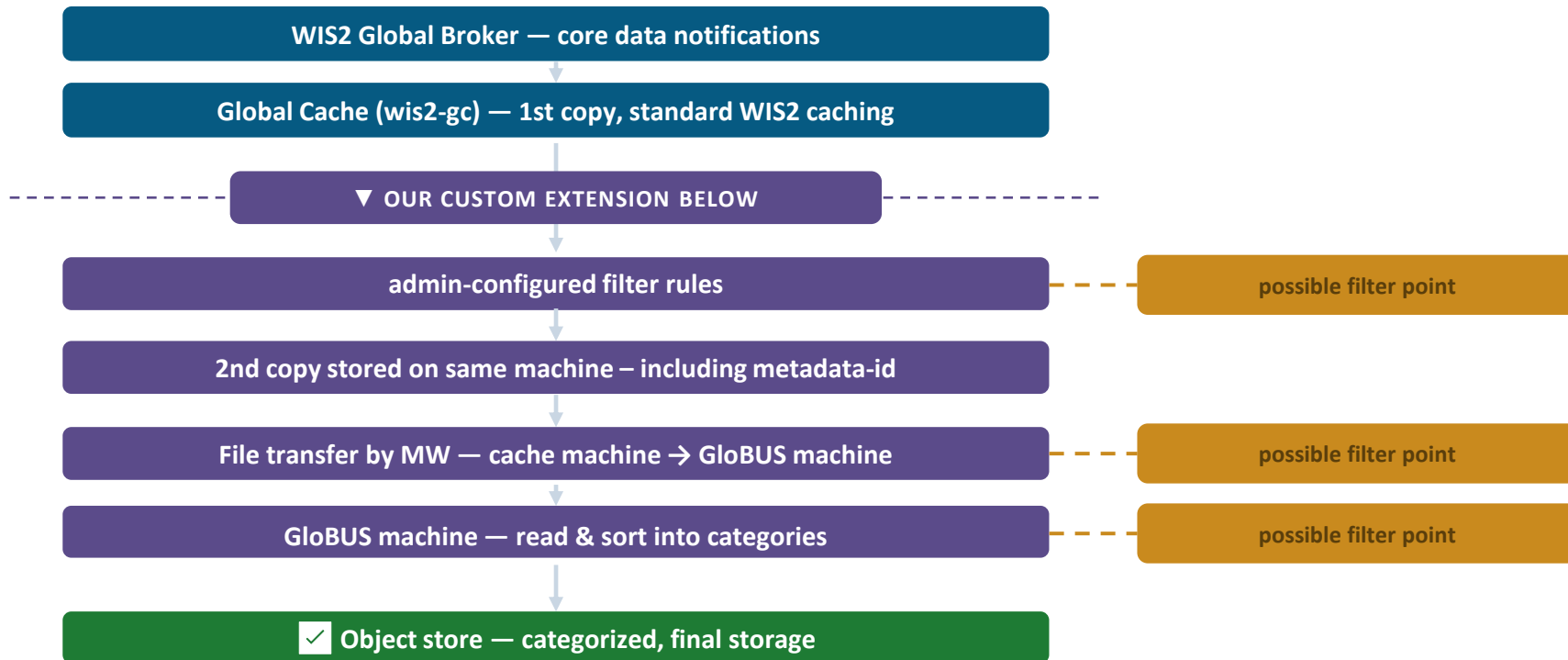
Global Cache @ DWD



Datavolume in GC Germany



From GC to wis2downloader



Where Should Filtering Happen?

Stage	Pros	Cons
In GC (source, our approach)	• Lowest bandwidth & storage — unwanted data is never fetched • Less load on every downstream stage • Filter criteria map directly to WIS2 concepts already present (topic, WNM properties, metadata-id)	• Rules are baked into the GC • Changing a rule doesn't recover data already skipped — no replay without re-subscribing • Harder to audit filter decisions separately from download/storage logic
During file transfer	• Decouples download from filtering • One place to add other logic too (validation, renaming) • Download stage stays simple and generic	• Data is already downloaded (and stored once) before being dropped — wasted work • Couples transfer jobs to business rules they otherwise wouldn't need
Directly before storage	• Full context available — metadata AND file content, not just topic/properties • Filter logic sits next to categorization logic — one place to reason about both • Upstream stages (download, transfer) stay generic and reusable	• Highest resource cost — download, storage and transfer still happen for data that's ultimately discarded • At high volumes, this cost is real and adds up

GC Admin Interface

WIS2 GC AdminadminChange PasswordLogout

[Filter Rules](#)[Auth Servers](#)[Users](#)

Storage Filter RulesImport RuleAdd Rule

Rules are evaluated top-to-bottom, first match wins. Use the arrows to reorder.

▲

▼

Synop CACHE + INTERNAL

CopyEditDisableDelete

Topics: `+/a/wis2/+/data/core/weather/surface-based-observations/synop`

Filter possibilities

→ Similar to wis2downloader:

- Centre-id
- Metadata-id
- Topic
- Content type

Add Filter Rule

Rule Name *

Target *

Cache + keep an internal copy

Cache + keep an internal copy

Cache only (skip internal copy)

Centre IDs (comma-separated, * wildcards)

e.g. fr-meteofrance, de-dwd-*

Metadata IDs (comma-separated, * wildcards)

e.g. urn:wmo:md:*

MQTT Topics (comma-separated, MQTT wildcards +/#)

e.g. origin/a/wis2/+data/recommended/#

Content Types (comma-separated, extensions or MIME)

e.g. .bufr, .grib2, application/x-bufr

Internal Folder (optional)

e.g. synop or /synop

Extra folder for matching files in internal storage. **synop** nests it inside the current folder (centre/synop/); **/synop** makes it top-level (synop/centre/). Cache paths and filenames are unaffected, and only new files move — existing ones stay where they are.

☒ Enabled

Cancel

Add

What about recommended data?

- Our GC is subscribed also to recommended topic
 - No filter: all messages are dropped
 - With filter: Same data flow as core data
 - Possibility to store credentials for specific URLs

Add Auth Server

Server Name *

Auth Type *

HTTP Basic

URL Patterns * (one per line, glob wildcards supported)

https://example.com/data/*
https://cdn.example.com/*

Username *

Password *

Cancel

Add

File transfer

- ➔ Using MovingWeather
 - ➔ Could be replaced by any other file transfer application (AFD, rsync, kafka, etc.)
 - ➔ File relay channel
 - ➔ Data is not touched, only relayed

Sorting (GloBUS)

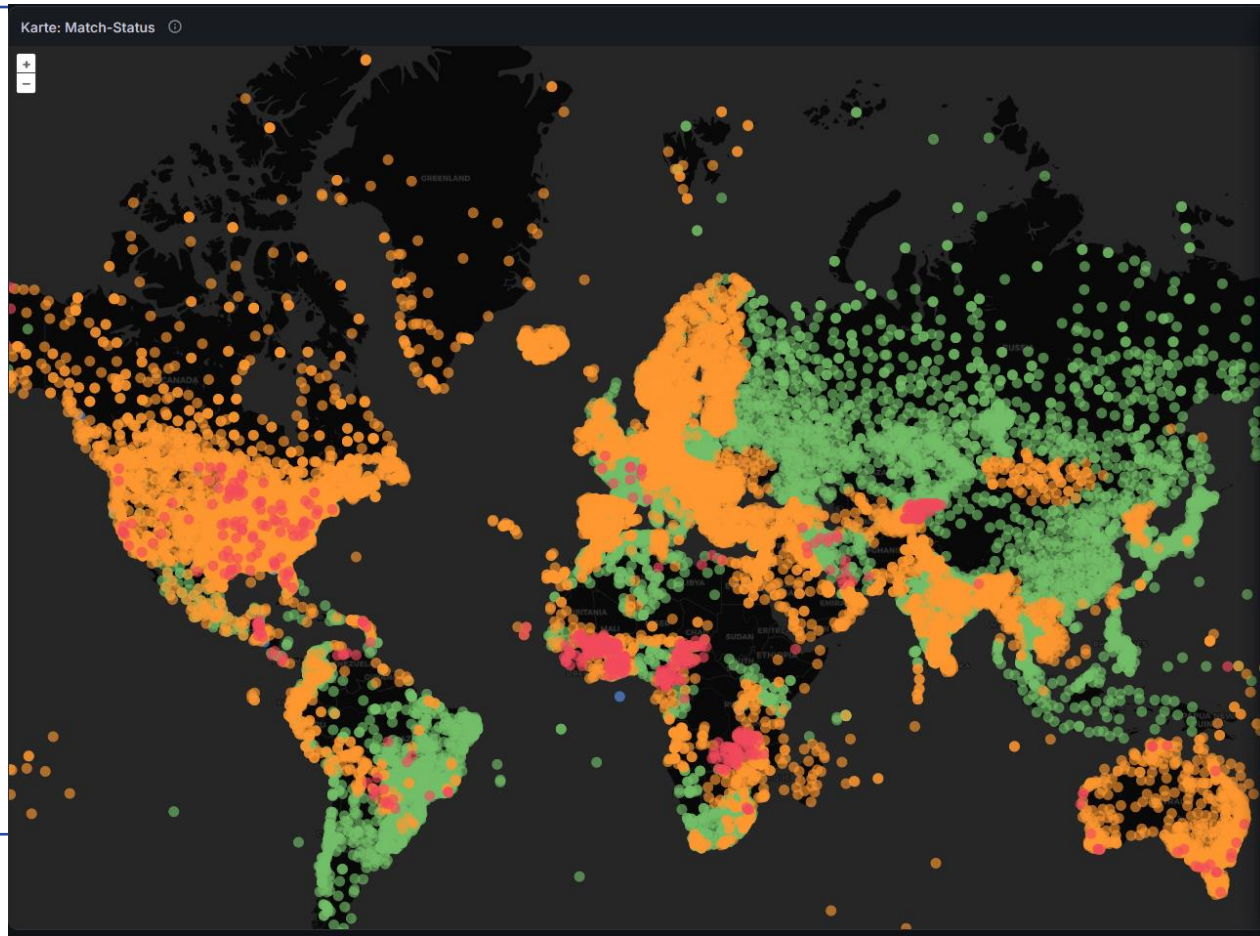
- ➔ Data is read from disk and is decoded
 - ➔ Decoding kernel for different file formats: GRIB, BUFR, ...
- ➔ (Station) metadata is extracted
- ➔ GloBUS decides via mapping in which database category data should be stored
 - ➔ Stores files + metadata
- ➔ Some internal routines rely on GTS Headers (metadata field)
 - ➔ Store metadata-id for downloaded WIS2 files
 - ➔ Use metadata-id as replacement for GTS Header (WIP)

WIS2 data in production

- Ship data
- Data from ng-nimet, ss-ssms, rw-rma, sc-seychelles-met, zw-msd, na-meteona

Synop Data

- GTS & WIS2
- Nur GTS
- Nur WIS2



Next steps

- ➔ Add source (GTS, WIS2, ...) as metadata field in objectstore
- ➔ Add more WIS2 data to production system
- ➔ Get notification about new WIS2 datasets

Thank you for your attention!