Refactoring data pipelines using containerization and continuous integration

Benjamin Bruns [*†]

[*]Institute for Advanced Simulation 8, Forschungszentrum Jülich, 52425 Jülich, Germany
[†] Institute of Bio- and Geosciences 2, Forschungszentrum Jülich, 52425 Jülich, Germany

Problem

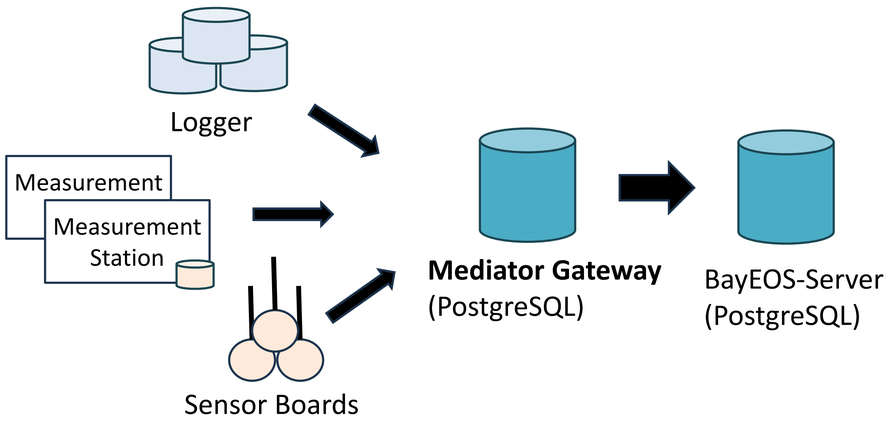
- Several processes (sensors, monitoring, automated analysis) generate different streams ([1], [2]) of time series (like) data
- Differences in data retrieval, data transformation and data aggregation
- Error-prone usage of standalone scripts for import into central data platform (BayEOS-Server, [3])

Motivation

- Idea to provide a flexible import and transformation pipeline as a template for all imports into existing backend (Bayeos-Server)
- Separation of responsibilities: distinct agents for import, transformation and aggregation processes
- Usage of containerization for data pipeline to ensure easy adaption, execution and traceability of pipeline processes

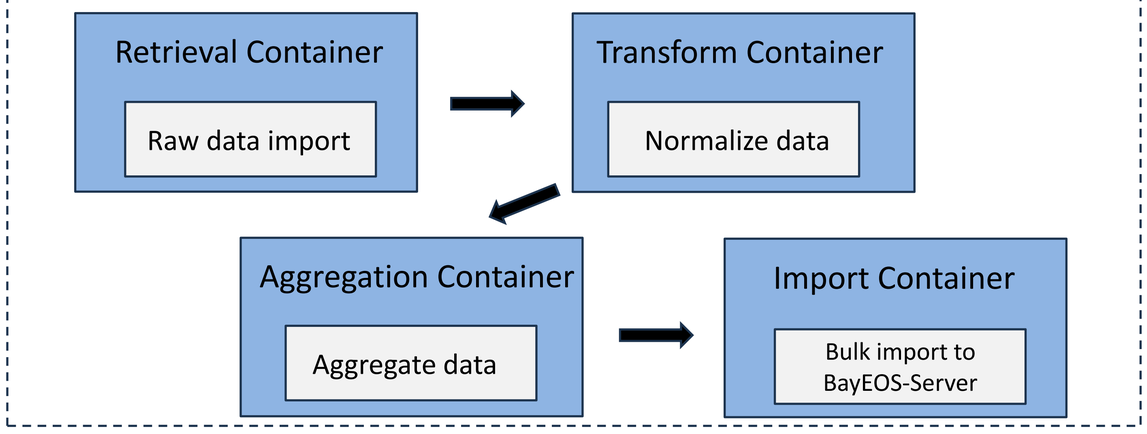
Pipeline architecture

- Introduction of "**Mediator Gateway**" to trigger data pipeline for each data source
- Mediator gateway runs as a container and caches (raw and transformed) data for bulk import to central data platform
- Different options / interfaces provided by Mediator Gateway to trigger data transfer:
  - (scheduled) push
  - scheduled pull
  - sample number limit (push via datalog)



Improved system architecture: Mediator Gateway

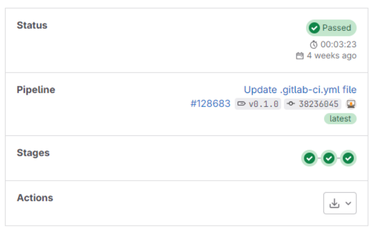
- Each pipeline step runs inside a distinct container:
  - Retrieval (push / pull): REST, (PL/pg)SQL, pg_cron
  - Transformation: (PL/pg)SQL, INSERT trigger
  - Aggregation: (PL/pg)SQL
  - Bulk import: (PL/pg)SQL, postgres_fdw
- Gitlab project templates (& documentation) for each pipeline step
- Repository includes CI/CD specification for automated building, tagging and deployment (to Gitlab registry) of containers



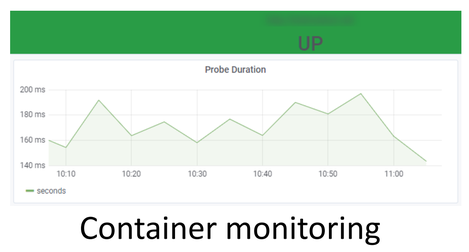
Containerized data pipeline

Refactoring: Benefits

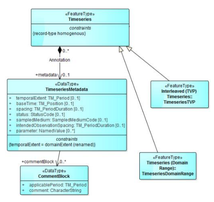
- Maintenance of pipeline (steps) is significantly simplified
- Inherent relationship (tag) between container and code repo
- Rollout and load balancing simplified by CI/CD automation and containerization
- More robust operation (restart policies) and simplified monitoring (Prometheus) due to containerization
- Fast bulk import (Mediator Gateway) speeds up data import
- Data harmonization on production side imposed by data schema of Mediator Gateway (simplified version of OGC O&M standard)



Automated build of a container



Container monitoring



Unified schema (simplified) of Mediator Gateway

Summary & Outlook

- Refactored data pipelines used in production for diverse data sources (mainly dense time series data)
- Containerization and CI/CD automation permits scalable operation of pipeline system and simplifies modifications
- Planned extension to generated Prometheus based monitoring system for data consistency checks

References

[1] Schmidt, F., et. al.: *A Distributed Information System For Managing Phenotyping Mass Data*, Referate der 33. GIL-Jahrestagung, Potsdam, 2013

[2] Bruns, B.., et. al.: *Entwicklung eines serviceorientierten Informationssystems für Phänotypisierungsmessungen im Freiland,* Geisenheim, 2015 Referate der 36. GIL-Jahrestagung, Osnabrück, 2016

[3] BayEOS-Server, Bayreuth Center for Ecology and Environmental Research, Universität Bayreuth, Code (LGPL): https://github.com/BayCEER/bayeos-server