



Spatial data processing with workflow engines

Pirmin Kalberer
@implgeo
Sourcepole AG, Zurich
www.sourcepole.com





Spacial data processing

‣ Typical tasks

- Data updates / imports
- Data exports / publication
- Raster processing
- Tile cache updates

‣ Classic tools

- Scripts (GDAL CLI, Python, ...)

‣ Orchestration

- Cron Jobs
- Makefiles
- Custom Scripts
- DB controled processes



Challenges

- Maintenance (many scripts, different backends)
- Execution overview (many jobs)
- Distributed processing (multiple servers)
- Containerization (dependencies)
- Log aggregation and analysis
- Process Monitoring with alerting (failures)



Data science → Tools!

awesome-workflow-engines

A curated list of awesome open source workflow engines

Full fledged product

- [Airflow](#) stars 27k - Python-based platform for running directed acyclic graphs (DAGs) of tasks
- [Argo Workflows](#) stars 12k - Open source container-native workflow engine for getting work done on Kubernetes
- [Azkaban](#) stars 4.1k - Batch workflow job scheduler created at LinkedIn to run Hadoop jobs.
- [Brigade](#) stars 2.4k - Brigade is a tool for running scriptable, automated tasks in the cloud — as part of your Kubernetes cluster.
- [CabloyJS](#) stars 633 - A Node.js full-stack framework with workflow engine, based on koa + egg + vue + framework7.
- [Cadence](#) stars 6.2k - An orchestration engine to execute asynchronous long-running business logic developed by Uber Engineering.
- [Camunda](#) stars 2.8k - BPMN-based workflow engine that can be embedded as java library (e.g. Spring Boot) or used standalone, including a graphical modeler and operations tooling.
- [CGraph](#) stars 328 - A simple-used and cross-platform DAG framework based on C++17 without any 3rd-party.
- [CloudSlang](#) stars 220 - Workflow engine to automate your DevOps use cases.
- [Conductor](#) stars 4.7k - Netflix's Conductor is an orchestration engine that runs in the cloud.
- [Copper](#) stars 220 - A high performance Java workflow engine.
- [Couler](#) stars 754 - Unified interface for constructing and managing workflows on different workflow engines, such as Argo Workflows, Tekton Pipelines, and Apache Airflow.
- [Covalent](#) stars 137 - Workflow orchestration platform for quantum and high performance computing.
- [Cromwell](#) stars 828 - Workflow engine written in Scala and designed for simplicity and scalability. Executes workflows written in [WDL](#) or [CWL](#).



Apache Airflow – DAG (directed acyclic graph)

Airflow DAGs Data Profiling ▾ Browse ▾ Admin ▾ Docs ▾ About ▾ 2018-09-07 22:15:40 UTC ⌂

On DAG: example_branch_dop_operator_v3 schedule: */1 * * * *

Graph View Tree View Task Duration Task Tries Landing Times Gantt Details Code Refresh Delete

Base date: 2018-09-05 01:04:00 Number of runs: 25 Go

BranchPythonOperator DummyOperator

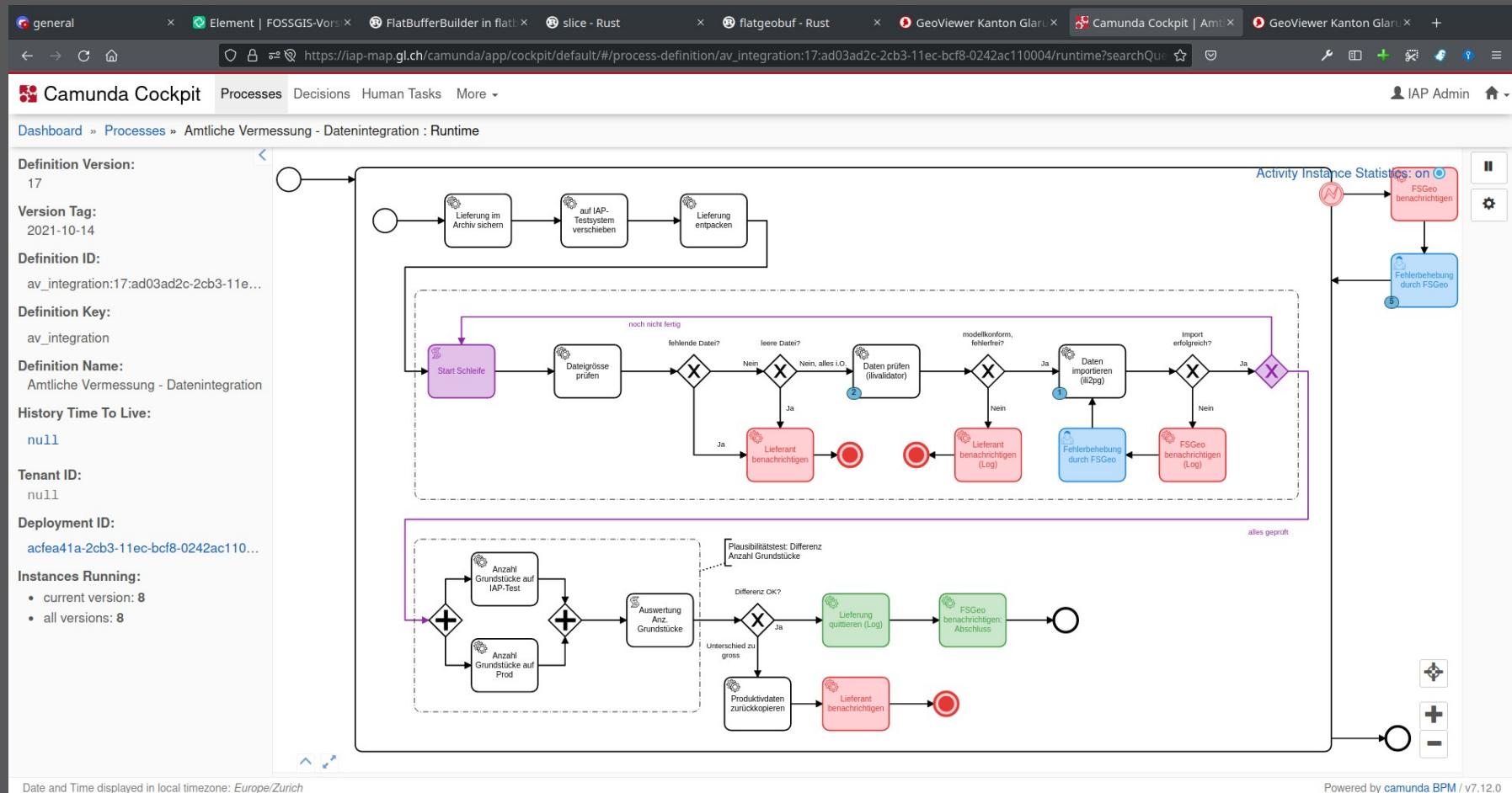
success running failed skipped retry queued no status

[DAG] oper_1 condition oper_2 condition

05:45 06 PM



Camunda - BPMN-based workflow engine





Dagster – multi backend processing engine

≡ ⚡ Search... / Runs Assets Status 🚨 Workspace ⚙

diamond Job in __repository__diamond@complex_pipeline.py

Overview Launchpad Runs

Select an op... ▾

Highlight...

Info Types

Description

No description provided

Resources

default

io_manager

The default io manager for Jobs. Uses filesystem but switches to in-memory when invoked through execute_in_process.

Any

console

The default colored console logger.

{
 log_level?: String
 name?: String
}

-o download_cereals

cereals Any

-o find_highest_calorie_cereal

cereals Any

-o find_highest_protein_cereal

most_calories Any

most_protein Any

-o display_results

Any

Type an op subset... (ex: download_cereals+)

The screenshot shows the Dagster UI interface. At the top, there's a navigation bar with icons for runs, assets, status, workspace, and settings. Below it, a job named 'diamond' is selected from a repository. The main area has tabs for 'Overview', 'Launchpad', and 'Runs', with 'Overview' currently active. A search bar at the top right includes a 'Highlight...' button. On the left, a dropdown menu says 'Select an op... ▾'. In the center, a pipeline graph is displayed. It starts with an operation 'download_cereals' (with an 'Any' constraint) which branches into two parallel operations: 'find_highest_calorie_cereal' and 'find_highest_protein_cereal', each also with an 'Any' constraint. These two operations then merge into a final operation 'display_results' (with an 'Any' constraint). Each operation is represented by a box containing its name and constraints. Below the graph, there's a search bar with placeholder text 'Type an op subset... (ex: download_cereals+)'. To the right of the graph, there's a sidebar titled 'Info Types' with sections for 'Description' (containing 'No description provided') and 'Resources'. Under 'Resources', there are entries for 'default', 'io_manager', and 'console'. The 'io_manager' entry provides a detailed description of the default file-based manager that switches to memory when using execute_in_process. The 'console' entry describes the default colored logger. At the bottom right, there's a small circular icon with a purple gradient.



Dagster – log collection

Search... /

Runs Assets Status Workspace ⚙️

Workspace 7918acbc Success Run of export_fixpunktprotokoll @ bfe470bc protokoll_pdf

22. Aug., 15:35:24 4,239s

Open in Launchpad View tags and config

create_db_schema_qwc
export_dm01av
export_fixpunktprotokoll
export_fpds2_ili
export_fpds2_to_csv
import_dm01av_testdata
import_fpds2_bilder
import_fpds2_from_csv
municipalities
pos_info
setup_fpds2
setup_fpds2_views
upload_to_checkch
user_query
validate_as_fpds1v95
write_fixpunkte_bilder_table

geo_processing_core_repository 9

2 of 2 shown Filter

Hide not started steps Re-execute all (*)

Not executed (0)
No steps are waiting to execute

Executing (0)
No steps are executing

Errored (0)

Type a step subset (ex: pdf_report+)

Hide unselected steps

Filter... debug 4 info 6 warning 0 error 0 critical 0 event 19

Copy URL

OP	EVENT TYPE	INFO	TIMESTAMP
pdf_report	RESOURCE_INIT_START	Starting initialization of resources [database_config, io_manager].	
pdf_report	RESOURCE_INIT_SUCC	Finished initialization of resources [database_config, io_manager].	15:35:26,801
pdf_report	LOGS_CAPTURED	Started capturing logs for step: pdf_report. captured_logs View stdout / stderr	15:35:26,852
pdf_report	STEP_START	Started execution of step "pdf_report".	15:35:26,863
pdf_report	STEP_INPUT	Got input "fixpunkte" of type "[String]". (Type check passed).	15:35:26,875
pdf_report	STEP_INPUT	Got input "regenerate" of type "Bool". (Type check passed).	15:35:26,883
pdf_report	INFO	JSON query: https://api3.geo.admin.ch/rest/services/api/MapServer/identify?geometryType=esriGeometryPoint&geometry=2678928.508,1243958.316&sr=2056&tolerance=0&layers=all:ch.swisstopo.pixelkarte-pk25.metadata&returnGeometry=false {}	15:35:27,027
pdf_report	INFO	Transformed response: 1091	15:35:27,149



Dagster – execution overview

≡ Search... /

Runs Assets Status Workspace ⚙

Workspace

- fpds2_processing_repository 16
- create_db_schema_qwc
- export_dm01av
- export_fixpunktprotokoll
- export_fpds2_ili
- export_fpds2_to_csv
- import_dm01av_testdata
- import_fpds2_bilder
- import_fpds2_from_csv
- municipalities
- pos_info
- setup_fpds2
- setup_fpds2_views
- upload_to_checkch
- user_query
- validate_as_fpds1v95
- write_fixpunkte_bilder_table

geo_processing_core_repository 9

Instance status

Overview Daemons Schedules Sensors Backfills Configuration 0:02 ⌂

2 of 2 Repositories Filter by job name...

Timeline 1hr 6hr 12hr 24hr ⌂ Hide

	12 Uhr	13 Uhr	14 Uhr	15 Uhr	16 Uhr	17 Uhr	18 Uhr
export_fixpunktprotokoll				2	13		
import_fpds2_from_csv						1	
export_fpds2_to_csv						3	

3 jobs succeeded

Job	Trigger	Latest run	View run	⋮
export_fpds2_to_csv fpds2_processing_repository@fpds2_processing.repos	None	22. Aug., 16:14 ⏲ 3,093s	View run	⋮
export_fixpunktprotokoll fpds2_processing_repository@fpds2_processing.repos	None	22. Aug., 15:35 ⏲ 4,239s	View run	⋮
import_fpds2_from_csv fpds2_processing_repository@fpds2_processing.repos	None	22. Aug., 15:33 ⏲ 3,386s	View run	⋮



Dagster backend libraries

- **Processing libraries / backends**
 - Pandas
 - dbt
 - Spark
- **Runtime environments**
 - Python
 - Celery, Dask
 - Docker, Kubernetes
- **API**
 - Python
 - GraphQL



› <https://ogcapi.ogc.org/processes/>



OGC API - PROCESSES

The OGC API - Processes standard supports the wrapping of computational tasks into executable processes that can be offered by a server through a Web API and be invoked by a client application. The standard specifies a processing interface to communicate over a RESTful protocol using JavaScript Object Notation (JSON) encodings. Typically, these processes execute well-defined algorithms that ingest vector and/or coverage data to produce new datasets.

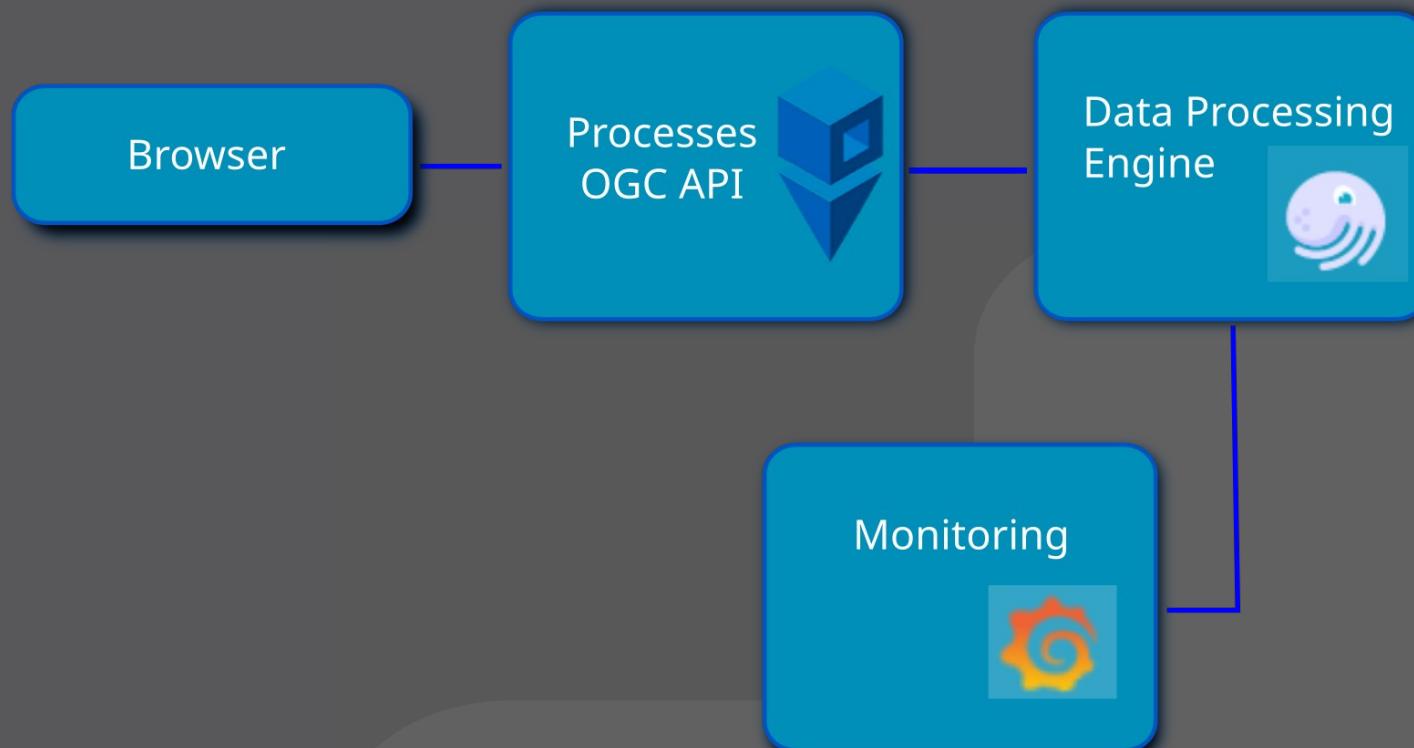
Overview

- Visit the [overview page](#) to learn more about this API.

Developer-friendly OpenAPI definitions



Processing engine + OGC API + Monitoring





Thank you!



Pirmin Kalberer
@implgeo