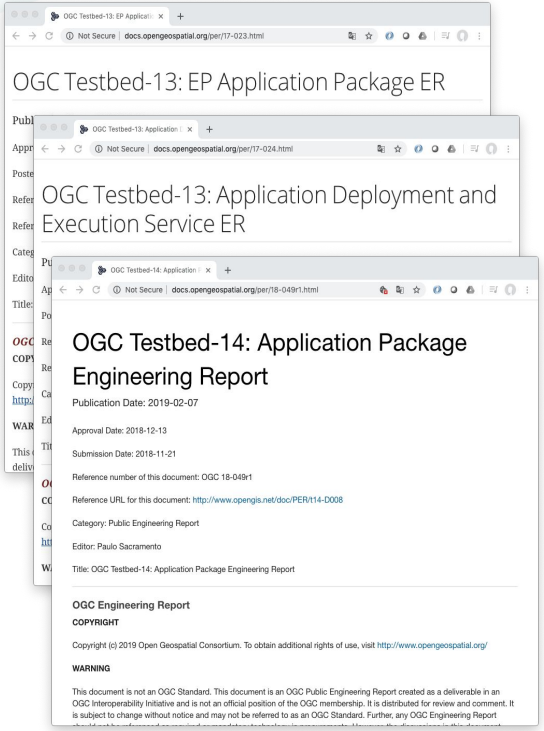
Processing & Chaining: Applications Deployment and Execution

Fabrice Brito
Terradue

ESA EO Φ-WEEK 2021 - EOEPCA
Friday - 15th October 2021

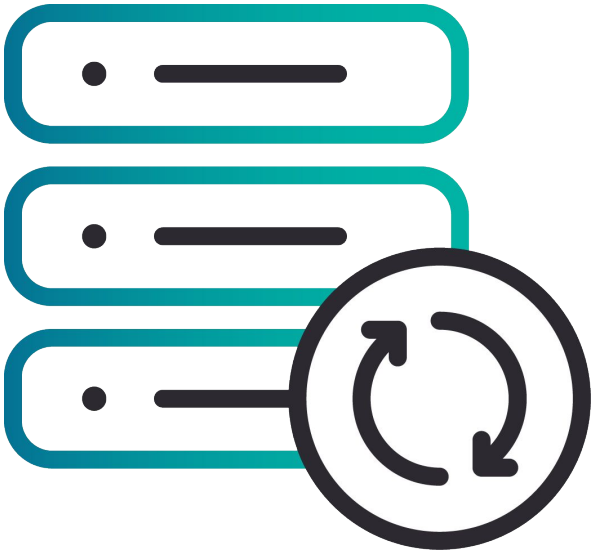
Earth Observation Application Package



- Previous OGC Testbeds 13-16 initiated the design of an *application package* for Earth Observation Applications in distributed Cloud Platforms
- The *application package* provides information about the software item, metadata and dependencies
- Deployed and executed within an Exploitation Platform in a service compliant with the OGC API Processes specification

By design, the Application Package defined during these testbeds targeted the deployment and execution using the OGC API Processes

Application Package Drivers



- Decouple application developers from exploitation platform operators and from application consumers:
  - Focus on application development by **minimizing platform specific particularities**
  - Make their **applications compatible** with **several execution scenarios**
- Enable exploitation platforms to virtually support any type of packaged EO application

Application Package Role

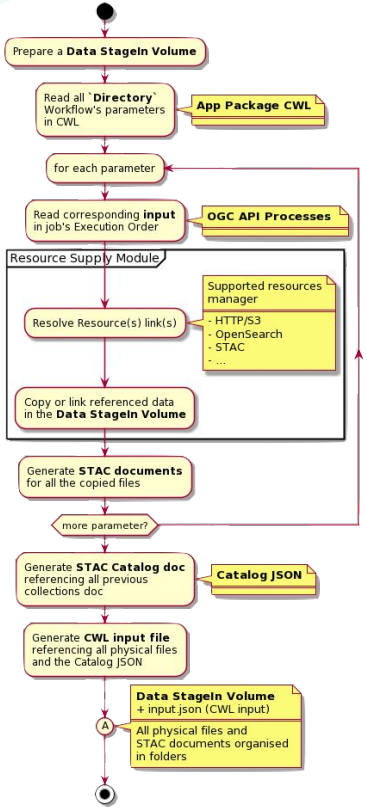
- Describes the data processing application by providing information about **parameters, software item, executable, dependencies and metadata**.
- Ensures that the application is **fully portable** supporting execution and automatic deployment in a Machine-To- Machine (M2M) scenario and **execution in other scenarios**.
- Contains an information model to allow its **deployment of the application as OGC API - Processes** compliant web service

EO Application Package Development

The Common Workflow Language (CWL) is an *open standard* for describing analysis workflows and tools in a way that makes them *portable* and *scalable* across a variety of *software and hardware environments*, from workstations to cluster, cloud, and high performance computing environments.

- Application developers create **containers** with their **runtime environment, dependencies and application binaries**
- Application developers:
  - Orchestrate the processing steps in a Directed Acyclic Graph (DAG)
  - Use fan-out or fan-in patterns at step level to exploit distributed computing resources
- The Application Package is described using the **Common Workflow Language (CWL)**

EO data flow management



- Use STAC as a data manifest for application inputs
- Use STAC as a data manifest for application outputs (metadata and results)
- Application developers have a clear method to consume EO data in their applications
- Platforms have a normalized and common specification for the data flow management

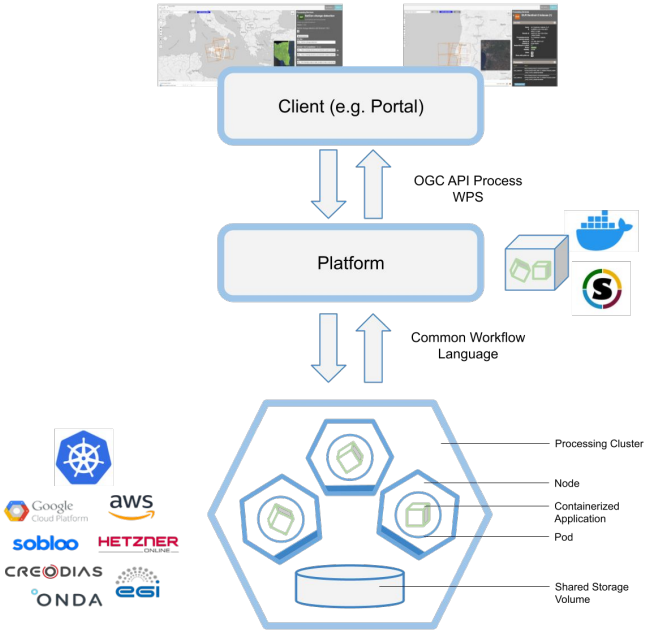
App. Pack. Development Environment

- The Processor Development Environment includes:
  - Integrated Development Environment (IDE): Theia
  - Exploratory tasks: Notebooks with JupyterLab
  - Container engine (e.g. docker or podman) for container build, pull, push
  - Link to Workspace (object storage, catalog, etc.)
  - Continuous Integration (CI) at repository level
- Supports the development and testing phases
- The Application Package is executed against reference data using the CWL reference runner, cwl-tool that:
  - Spawns one or more containers for each CWL step (node of the DAG)
  - Inputs, intermediary results and outputs are read/written in container volumes according to the CWL workflow

Execution on distributed processing environments - Kubernetes

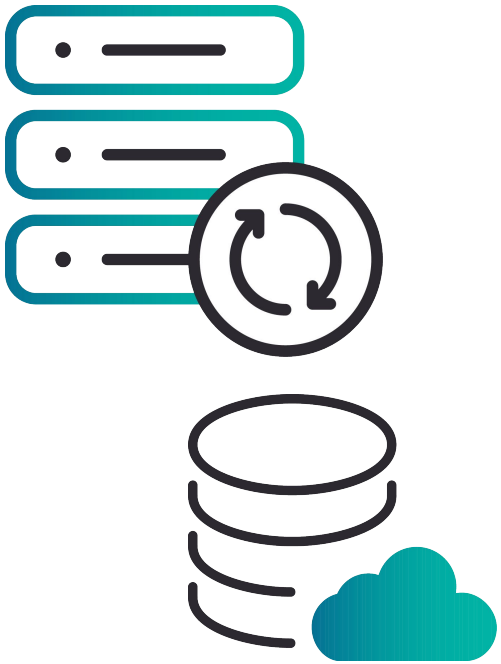
- The Application Package is a way to make software portable and executable on a Kubernetes cluster
- Kubernetes is an open-source container-orchestration system for automating computer application deployment, scaling, and management providing
  - Service discovery and load balancing
  - Storage orchestration
  - Automatic bin packing
  - Batch execution
  - Horizontal scaling
  - Self-healing
  - Automated rollouts and rollbacks
- Each CWL step (node of the DAG) spaws one or more kubernetes Pods
- Inputs, intermediary results and outputs are read/written by the spawned Pods according to the CWL workflow

Deployment and Execution on Exploitation Platforms



- The Application Package can be deployed on an Exploitation Platform:
  - The Application Package is deployed
  - An OGC API Processes processing service is exposed and ready to receive execution requests
- EOEPCA's implementation of the ADES:
  - The underlying CWL runner is Calrissian
  - Kubernetes provides the processing resources
  - The Zoo project provides the OGC API Processes

Conclusion



- ✅ Decouple application developers from exploitation platform operators and from application consumers
- ✅ Provide Cloud native EO applications development environment
- ✅ Provide multi-tenant, scalable and transient processing resources

Looking forward hearing from you!

https://www.terradue.com

Fabrice Brito

fabrice.brito@terradue.com

References

OGC Best Practice For Earth Observation Application Package
https://gitlab.ogc.org/ogc/eoap-best-practice/ (TC vote on-going)

Common Workflow Language
https://www.commonwl.org/

SpatioTemporal Asset Catalogs
https://stacspec.org/

CWL on Kubernetes - Calrissian
https://github.com/Duke-GCB/calrissian