Serverless Computing and FaaS for Airport Meteorology

Cybernetics & Informatics 2022

Dr. Martin Bobák

Institute of Informatics of the Slovak Academy of Sciences

September 13, 2022

INSTITUTE OF INFORMATICS
SLOVAK ACADEMY OF SCIENCES

Outline

Introduction

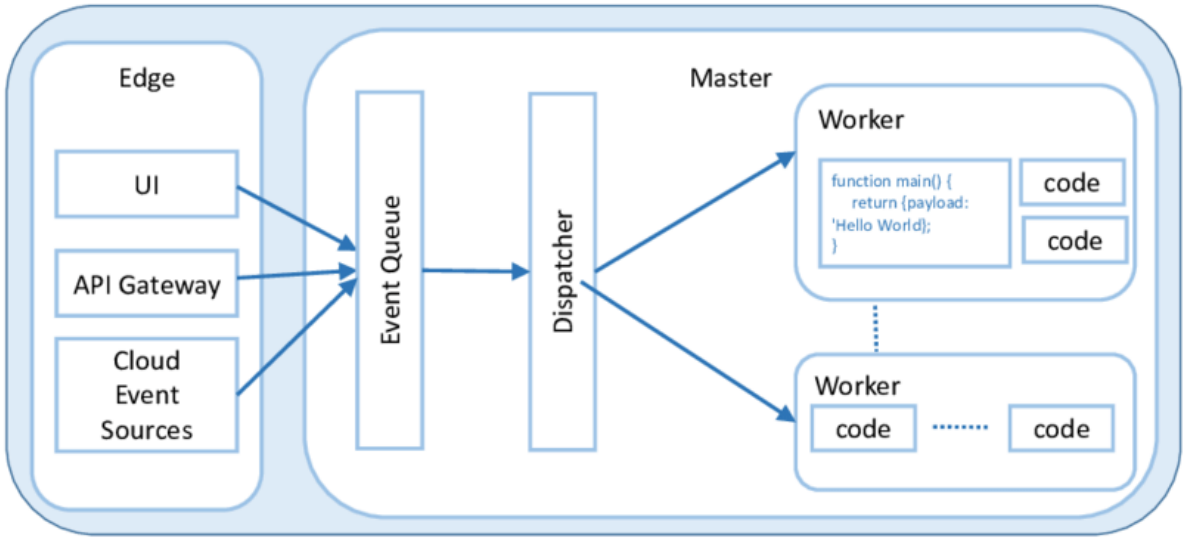
- We present the initial stages of the construction of an airport visibility meteorological application based on the Function-as-a-Service paradigm
- monolythic version already in use at airports
- transformation into a server-less application

Motivation

- For the research partner:
  - part of long-standing research in cloud computing
  - acquisition of new know-how
  - opportunity to verify research results in real life use
- For the application developer:
  - modularization of application, allowing several different deployments (different requirements, functionality, pricing)
  - unburdens customers from hardware acquisition and maintenance
  - easier development of new functionality, even spanning new domains
  - opportunity to gain know-how in modern computing paradigms

What is Function as a Service?

- Subset of serverless computing that provides a platform allowing developers to write and deploy applications without building and maintaining the underlying infrastructure
- infrastructure management (resource provisioning, maintenance and regular update of base operating systems) are the responsibility of the Cloud provider
- developer focuses only on applicaiton code and logic



Advantages of FaaS

- Automatic scaling: functions are scaled automatically, independently, and instantaneously according to the actual demands by the cloud provider. That relieves developers from concerns of high traffic or heavy use.
- Cost efficiency: Users have to pay only for the computing resources they really use, not for idle resources that are often reserved for handling possible high demands in the typical IaaS (scaling by cloud provider).
- Quick development: developers don't have to manage infrastructure, they can focus only on the code, reducing the cost of development and the time to market.

Disadvantages of FaaS

- Potential vendor lock-in: The application codes are built on the top of a concrete FaaS platform and difficult to port to another vendor.
- Difficulties for testing: The codes are running on the top of a FaaS platform, it may make difficulties for creating local test environments for applications

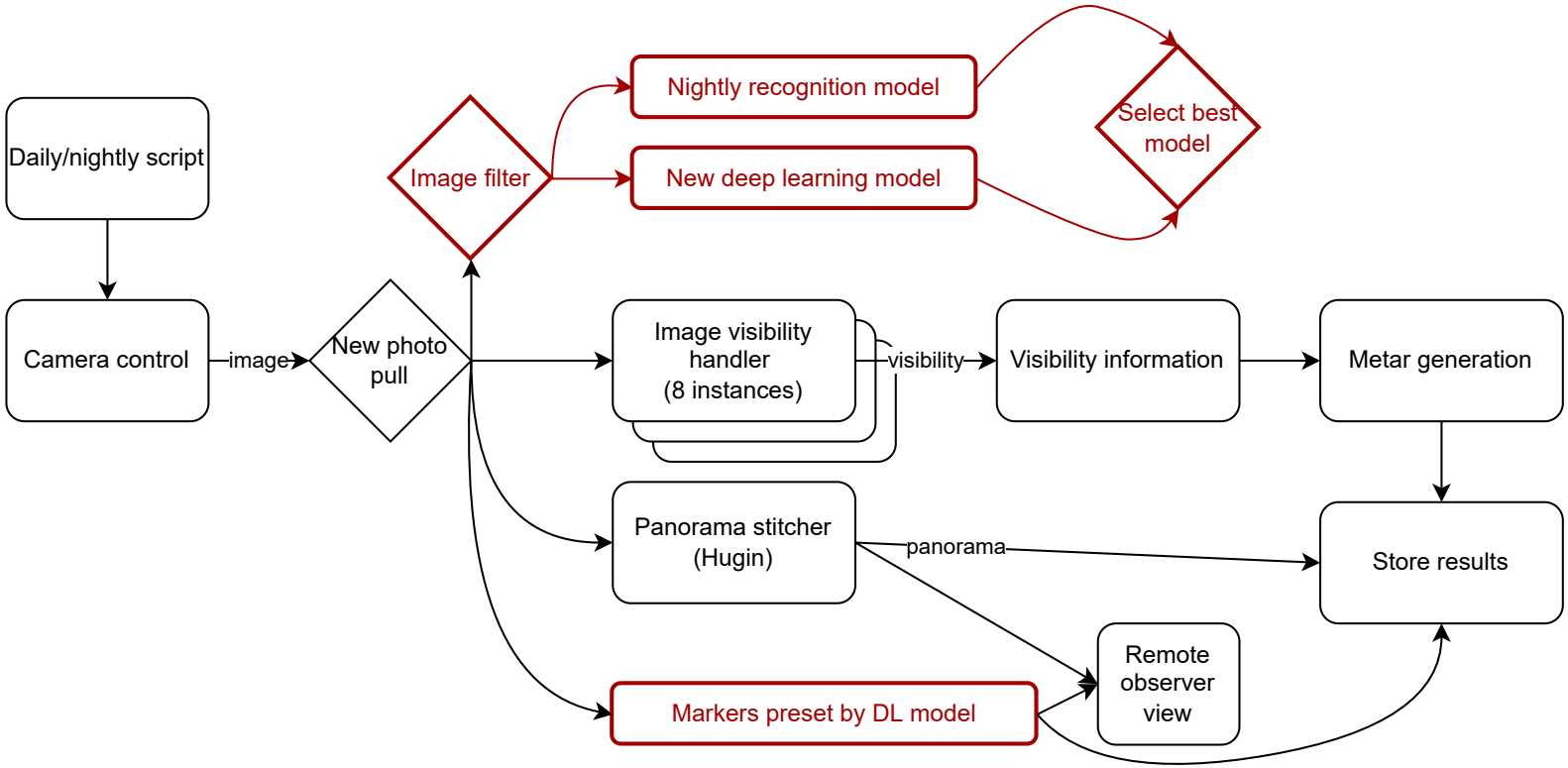
Current FaaS Frameworks

- The first commercial provider offering FaaS is Amazon AWS with **AWS Lambda** platform,
- followed by Google with **Google Cloud Functions**.
- We will focus on two open-source platforms: **Apache OpenWhisk**, originally by IBM,
- and **OpenFaaS** (by a company of the same name)

Application - Motivation

- Visibility is a crucial element in the safety of all kinds of transport
- Almost 50% of all aircraft accidents is due to weather conditions
- main cause of weather-related aviation accidents is reduced visibility
- better visibility information also leads to better traffic management
  - reduced fuel consumption
  - reduced flight delays

Application - Architecture

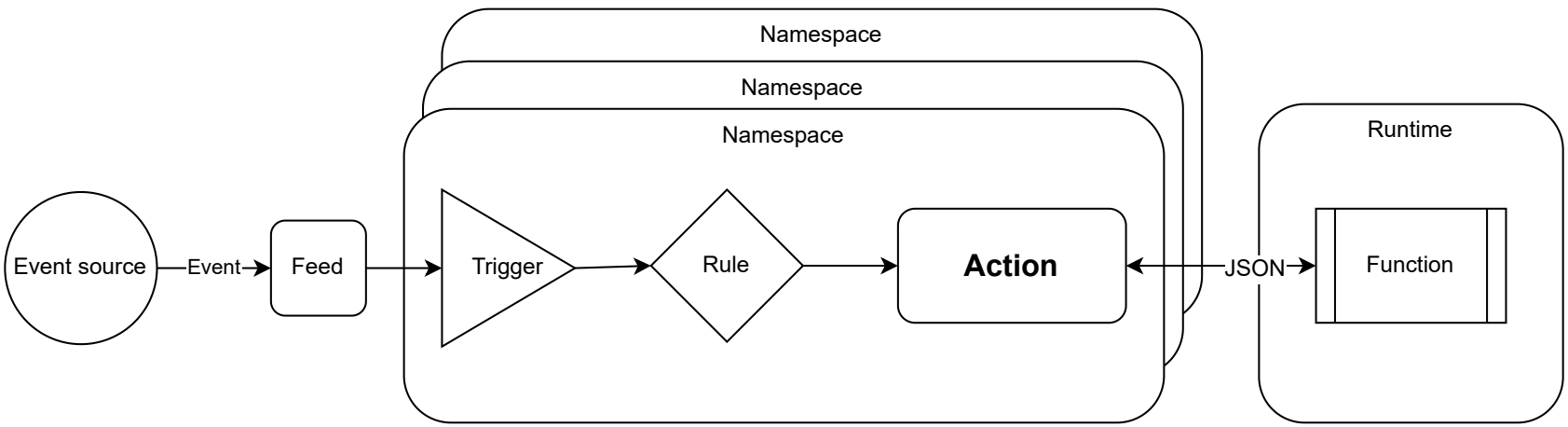


OpenWhisk - Origin and Use

- free and open implementation of the Function-as-a-Service paradigm
- tied to AWS Lambda service, first presented in November 2014
- started by Rodric Rabbah of IBM Research
- initially developed at IBM Research as Whisk
- later renamed to OpenWhisk, made OSS and transferred to the Apache Software Foundation Incubator
- can be installed in several ways, we have used Kubernetes
  - Helm chart for OpenWhisk is available
  - local installation of wsk command-line tool also necessary on the developer's machine

OpenWhisk - Programming Model

- an event-driven system
- event from an even source feeds into a trigger
- a trigger uses rules to execute an action
- uses REST API to accept new events
- accepts functions in several languages (Java, Python, PHP, Go, Ruby...) as well as black-box code



OpenWhisk - Application

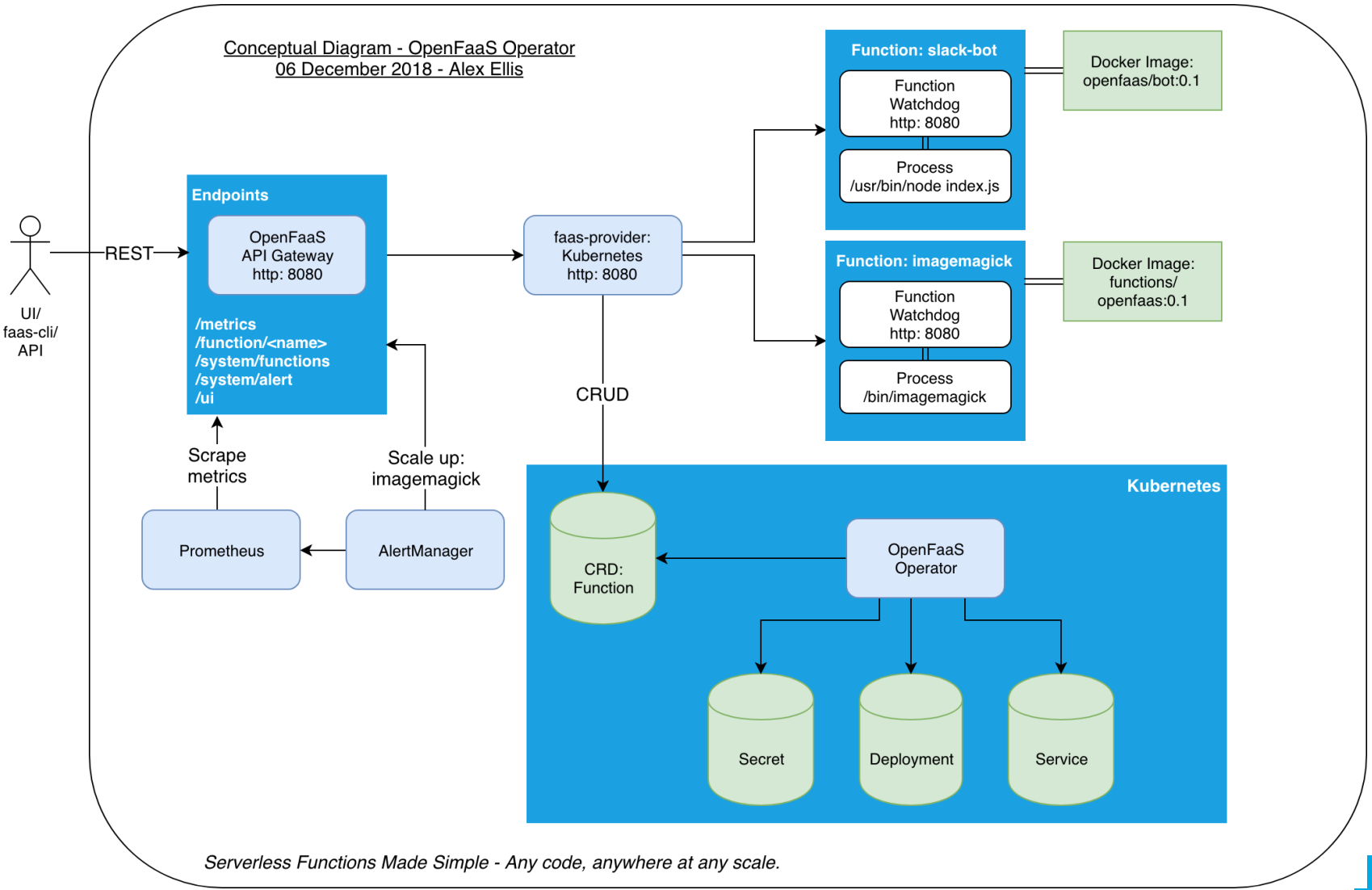
- application will be implemented according to the architecture shown above
- so far implemented:
  - ImageVisibilityHander, as a Java action
  - Visibility info from 8 xmls, also as a Java action
  - Panorama stitch, as a black-box docker action, since we use a 3rd party software (Hugin[1])
- In the case of the Panorama stitch action, we have created a specific docker image, based on OpenWhisk's Docker Runtime image, adding the Hugin software for panorama-stitching
- Panorama stitch could also be done as a Python action with a custom docker image for OpenWhisk's Python actions

[1] Hugin - Panorama Photo Stitcher, `https://hugin.sourceforge.io/`

OpenFaaS - Introduction

- framework for serverless computing
- written in the programming language Go
- uses a Docker container for running a serverless function in an isolated environment
- supports several ways of container lifecycle and networking management
- basic container orchestration is obtained by Kubernetes or Docker Swarm
- also possible to integrate with other orchestrators via FaaS Nomad
- OpenFaaS Pro is a commercially licensed distribution of OpenFaaS

OpenFaaS - Architecture



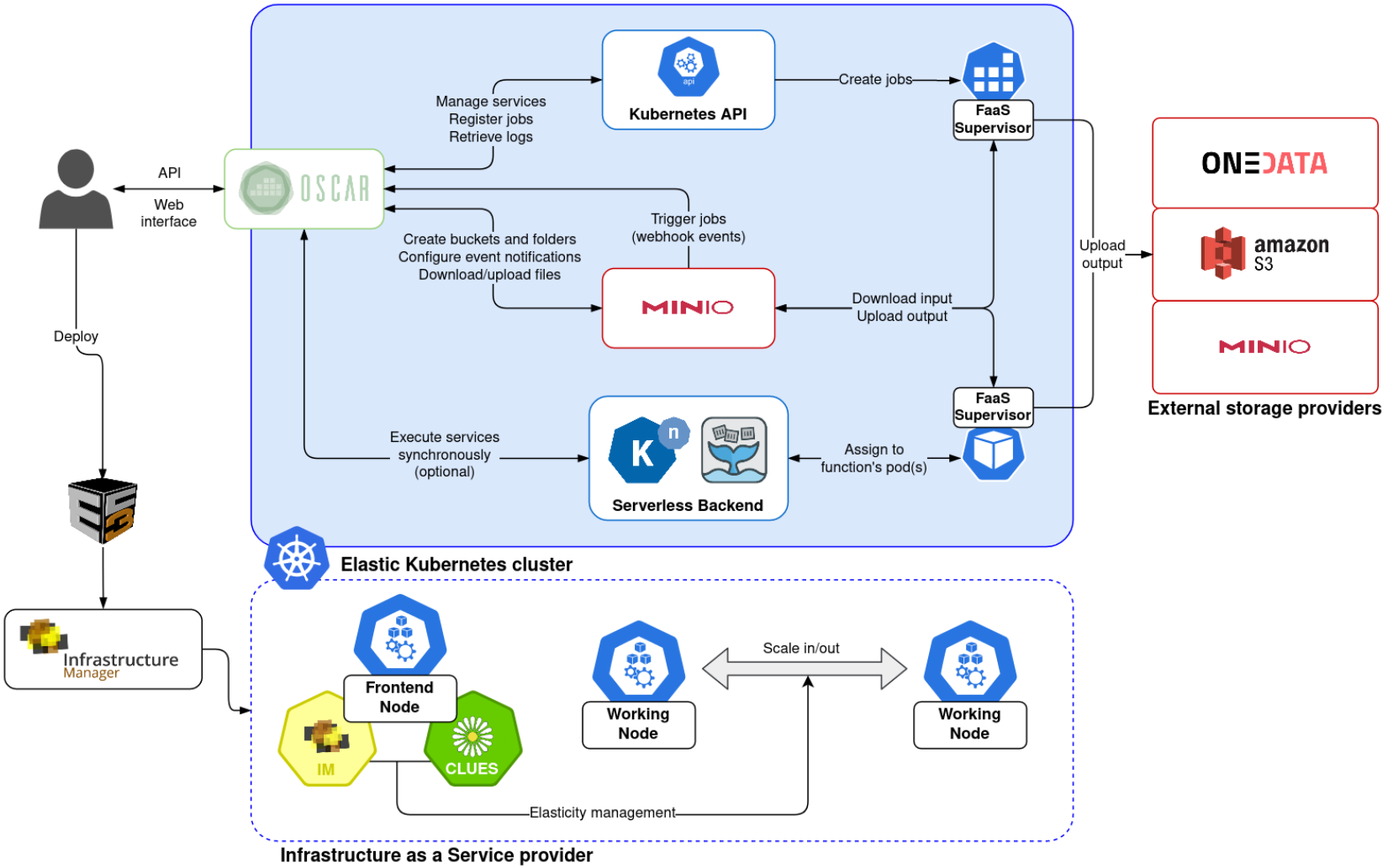
OpenFaaS - Programming Model

- utilizes an event-driven programming model
- functions are triggered by events, the most common events are HTTP requests
- it is also possible to have asynchronous functions
- an eventconnector pattern allows users to create a broker or a separate microservice which maps functions to topics and invokes functions via the OpenFaaS Gateway
- OpenFaaS Pro supports events coming from Apache Kafka, and AWS SQS.
- function deployment via OpenFaaS CLI
- OpenFaaS supports several programming languages (Java, Python, Go, Ruby, C#, PHP, Node.js)

OSCAR - Introduction

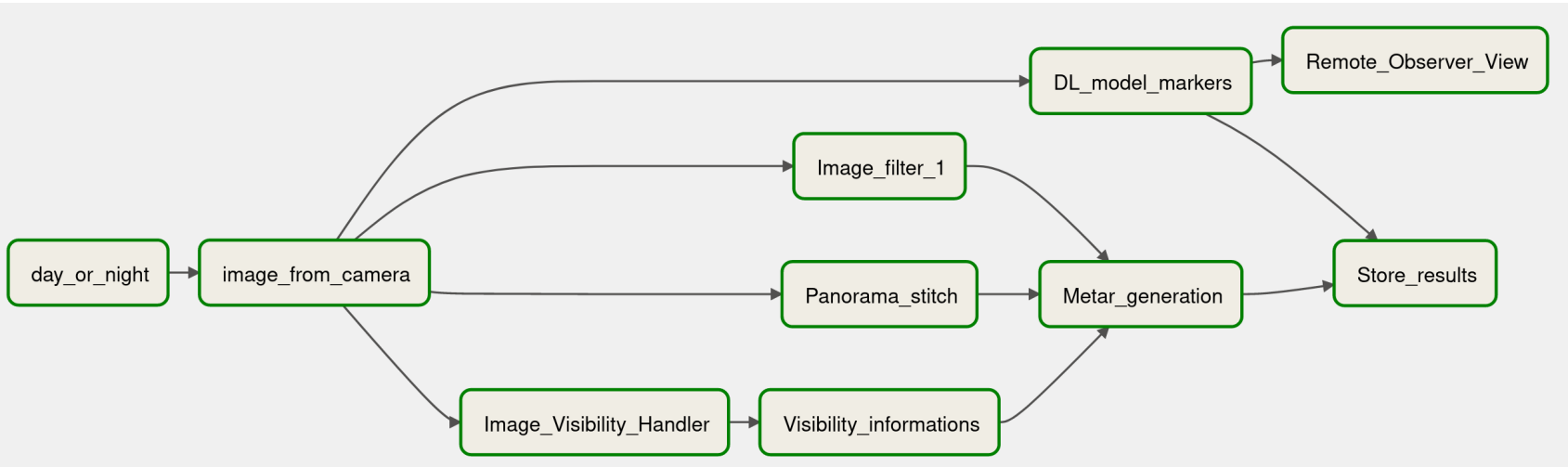
- open source framework for data-driven serverless computing
- designed for applications that are modular according to the incoming files
- built on OpenFaaS which provides automatic function deployment
- OSCAR offers an elastic multi-cloud environment managed by Kubernetes integrated with storage back-ends triggering serverless functions upon file uploads
- storage environment is designed to work with objects which are highly distributed
- access via REST API, web-based UI, command-line interface

OSCAR - Architecture



Airflow

- open source framework for lightweight serverless functions management
- amalgamates individual tasks into a workflow which is expressed as a directed acyclic graph
- in our case the tasks are serverless functions
- the resulting workflow characterizes the relations between its tasks which also defines their execution order.



Summary and Future Work

- initial work on the transformation of a monolithic application into the serverless cloud domain (FaaS)
- we have chosen the open source OpenWhisk platform
- this platform allows the event-driven execution of actions
- in our further work, we will continue to transform more modules of the original application into serverless functions
- we will to test as much of the functionality of OpenWhisk as possible
- we will test its throughput and responsiveness
- we will evaluate whether the ported application is usable for its developers and their customers