An Event-Driven Architecture Based on Copernicus Satellite Data for Water Monitoring

Sebastian Drost
Bochum University of Applied Sciences
Lennershofstr. 140
Bochum, Germany
sebastian.drost@hs-bochum.de

Matthes Rieke
52°North GmbH
Martin-Luther-King-Weg 24
Münster, Germany
m.rieke@52north.org

Simon Jirka
52°North GmbH
Martin-Luther-King-Weg 24
Münster, Germany
jirka@52north.org

Arne Vogt
Bochum University of Applied Sciences
Lennershofstr. 140
Bochum, Germany
arne.vogt@hs-bochum.de

Verena Rabea Kirstein
Wupperverband
Untere Lichtenplatzerstraße 100
Wuppertal, Germany
vkn@wupperverband.de

Andreas Wytzisk
Bochum University of Applied Sciences
Lennershofstr. 140
Bochum, Germany
andreas.wytzisk@hs-bochum.de

Abstract

Climate changes and the ongoing intensification of agriculture cause increased material inputs in watercourses and dams. Thus, water industry associations, suppliers and municipalities face new challenges. Geolocating and quantifying pollutant inputs in surface water bodies are particularly important tasks within an improved water monitoring program. Hydrological models are an essential building block for such a program, but they often suffer from the sparse coverage of in-situ data. Due to the evolvement of the Copernicus satellite platforms, the broader availability of satellite data provides a great potential for optimizing hydrological models. This paper presents a system architecture that is based on Copernicus offerings and integrates heterogeneous data sources and existing web-based information systems in order to provide extended earth observation analysis services that contributes to an improved water monitoring. We discuss the major challenges of this approach and propose an event-driven architecture that aims to exploit the potential of the different processing environments provided by the Copernicus data platforms.

Keywords: Water Management Monitoring, Copernicus Satellite Data, Event-Driven Architectures, Spatio-Temporal Data

1 Introduction

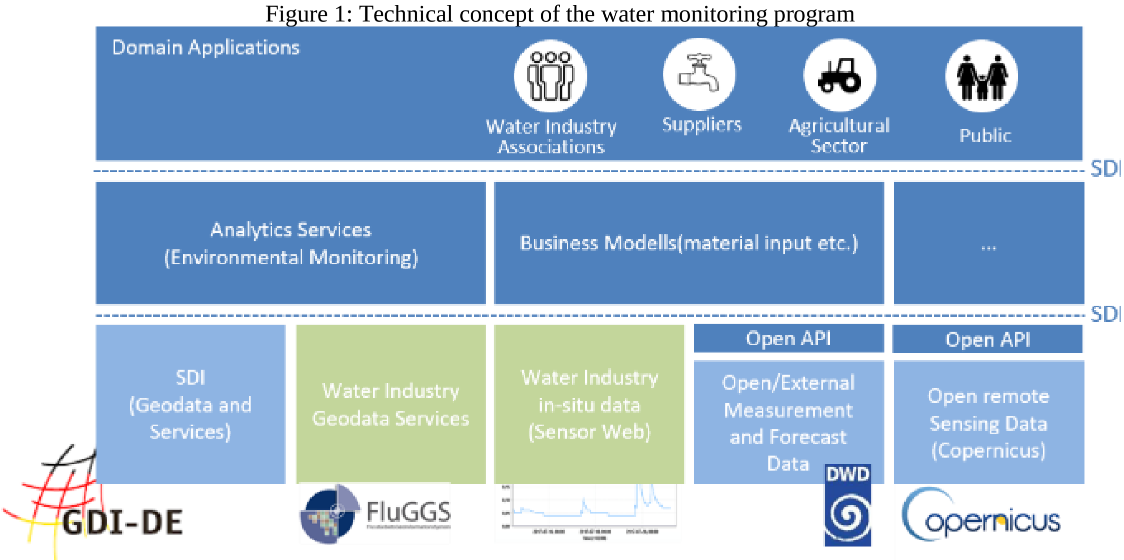
Due to climate changes and the intensification of agriculture, water industry associations, suppliers and municipalities face new challenges regarding the quality control of drinking water. Fertilization, heavy rains and soil erosion result in increased sediment and material inputs into watercourses and dams. In order to ensure an environment friendly water resource management, there is a great need for improved water monitoring programs as well as strategies to reduce pollutant inputs into surface waters. Geolocating and quantifying material inputs into surface water bodies are particularly important tasks contributing to this goal.

In the past, associations for water management primarily utilized in-situ data for small-scale hydrological models. Due to the limited coverage of highly accurate in-situ data, model outputs often suffer from large uncertainty (Seibert and Beven, 2009). With advances in global earth observation and the broader availability of remote sensing products, different approaches have demonstrated how to overcome the lack of suitable in-situ data by exploiting satellite data for different hydrological problems, such as soil moisture determination, evapotranspiration estimation, or streamflow prediction (López López et al., 2017).

Today's evolvement of Copernicus satellite data platforms such as the Copernicus Data and Exploitation Platform - Deutschland (CODE-DE), the Copernicus Open Access Hub and the Copernicus Data and Information Access Services (DIAS) provide easy access to Copernicus data, products and tools. They also enable access to infrastructures to build advanced applications for environmental monitoring.

Our research project aims to exploit the great potential of these platforms for the development of innovative services for water management analytics and the improvement of hydrological models in order to optimize monitoring processes. Existing SDI-based geodata and web-services as well as in-situ data from the Sensor Web will be combined with Copernicus Sentinel-1 and Sentinel-2 data. In addition, we will develop a solution for the connection to the Copernicus data platforms. This paper covers the different challenges associated with these goals. It presents a modular and extensible software architecture for water monitoring use cases that is based on interoperable Open Geospatial Consortium (OGC) interfaces. Architectural aspects addressing the need for an event-driven communication – i.a. publish/subscribe patterns and messaging protocols – are also addressed (Rieke et al., 2018).

Figure 1: Technical concept of the water monitoring program



2 Related Work

In order to build an efficient infrastructure that supports the water monitoring tasks outlined in the introduction, several related developments need to be taken into account. These are primarily the integration of in-situ observations as well as the provision and handling of remote Earth Observation data.

The Sensor Web Enablement (SWE) framework standards developed by the Open Geospatial Consortium (OGC) must be considered for accessing in-situ observation data. These standards provide a comprehensive suite of interface and data model/encoding specifications (Botts et al., 2006). The applicability of these standards for water monitoring tasks has already been proven in practice (Stasch et al., 2018).

Complementary to in-situ measurements, which are typically gathered by the water industry itself, the large amounts of the Copernicus program's raster-based Earth Observation data are an important resource for this work. The availability of data access and exploitation platforms such as CODE-DE (Storch et al., 2018) and the European DIAS nodes needs to be considered. Furthermore, there is a broad range of activities addressing the question how Copernicus can support water resource management (European Commission, 2017).

Finally, the OGC Publish/Subscribe standard provides a specification for enabling push-based communication patterns in spatial data infrastructures. Although it already provides basic functionality to deliver observation data as soon as it becomes available, the specification is currently only available as a SOAP/XML binding. Further work is necessary to support more lightweight communication flows (Bigagli and Rieke, 2017).

3 Challenges

An interoperable integration of remote sensing data and existing SDI-based geodata and services, in-situ data and open meteorological data will enable the development of innovative analytics services for water monitoring. Based on a standardized API, domain- and task-specific applications can be built for various user groups within the hydrological sector (Figure 1). Some viable use cases are e.g. the extraction of land cover dynamics and interannual variability of soil moisture, the recognition of macrophyte hotspots as well as the improvement of models describing pollutant flows.

There are various challenges to fulfilling requirements for a modular and interoperable software architecture based on different data sources and prepared to connect to Copernicus data platforms. These challenges apply to their integration into a water management association's existing IT-infrastructure

3.1 Triggering of process workflows

Depending on the domain specific requirements, different processing workflows have to be triggered either in fixed time intervals or as soon as new datasets are available. For instance, a land cover classification could be executed every time a new satellite scene for a certain area of interest is available, while the monthly aggregated soil moisture only has to be calculated once a month. Such an event-based process execution requires the discovery of different data sources in order to detect suitable spatio-temporal datasets. This task also has to consider the timely delivery of new datasets and the update intervals of existing ones.

3.2 Interoperable processing of large earth observation data

Common open-source tools for processing Earth Observation data are the Sentinel Toolboxes based on the Sentinel Application Platform architecture (SNAP), the Orfeo ToolBox, GRASS GIS or the Integrated Land and Water Information System (ILWIS). These tools are built on open-source, third-party libraries, such as GDAL or OpenCV. They provide either an API for one or more programming languages, a command line interface, or both. Since each of these tools focuses on different aspects of satellite imagery processing, a combined use offers a great potential for automatizing processing workflows. The challenge is to develop and establish a standardized and interoperable interface for executing such processes.

3.3 Data processing close to the data

While in-situ sensor data processing usually does not have significant impact on infrastructures, calculations on large Earth Observation (EO) data requires a large amount of storage and processing capability. It is necessary to execute the processes close to the data to minimize the network traffic caused by downloading and transferring large image data into a private execution environment. Some of the Copernicus data platforms such as DIAS web-services, CODE-DE or Sentinel on AWS (Amazon Web Services) provide a scalable computing and storage environment for deployment and execution of custom algorithms. Hence, there would be a significant benefit to exploiting these infrastructures.

4 Approach

4.1 Architecture Overview

The system architecture is designed in a modular fashion and follows a publish/subscribe pattern (Figure 2). The different components are loosely connected to each other via messages that are passed through a message broker. Each module subscribes to messages of interest at the message broker. This approach enables an independent and asynchronous handling of specific events. The different components and their interactions are described as follows.

Job Manager: The Job Manager provides a REST API to the domain related user groups in order to define job descriptions for planning the execution of analysis processes. These comprise conditions such as required input data, the area of interest, execution intervals as well as certain quality requirements (e.g. a threshold for cloud coverage of satellite imagery).

Core Engine: The job definitions will be evaluated by a Core Engine that schedules jobs for periodical process executions. In addition, it will react to the notifications about the availability of new data by triggering the execution of the corresponding data analysis processes.

Datasource Observer: A Datasource Observer runs several observing routines to request datastores for new available data, such as in-situ measurements or Copernicus satellite data. It subsequently publishes messages that contain the metadata for new datasets. Since the Datasource Observer holds a subscription to the creation of new job definitions, it will only look for those data sets that are actually needed for the process executions.

Data Wrapper: Information about all incoming required datasets are stored in a Metadata Storage. The Data Wrapper has access to this storage. For the purpose of defining process inputs, it generates references to the required datasets from the metadata and provides these references to the Core Engine via a REST API.

Web Processing Service: The execution of analysis processes provided by EO Tools is encapsulated by a OGC Web Processing Service (WPS), which provides a standardized interface for this purpose. A WPS process is triggered as soon as all inputs are available and the analysis process is ready for execution. Since the notification message for such an event only contains references to the input data, the WPS takes care of fetching and preparing the data to make it available in the processing environment.

As one can see, each component of our architecture encapsulates certain limited tasks. This enables the single components to be run as microservices which provide horizontally scaling capabilities for our system. We designed the systems' components in a way that facilitate its deployment as containers. The message broker plays an important role in this architecture, since it is responsible for routing messages between the single components' instances. Overall, such a microservice design prepares our system for operating in a cloud environment that builds on container virtualization.

Figure 2: Architecture for the water monitoring system

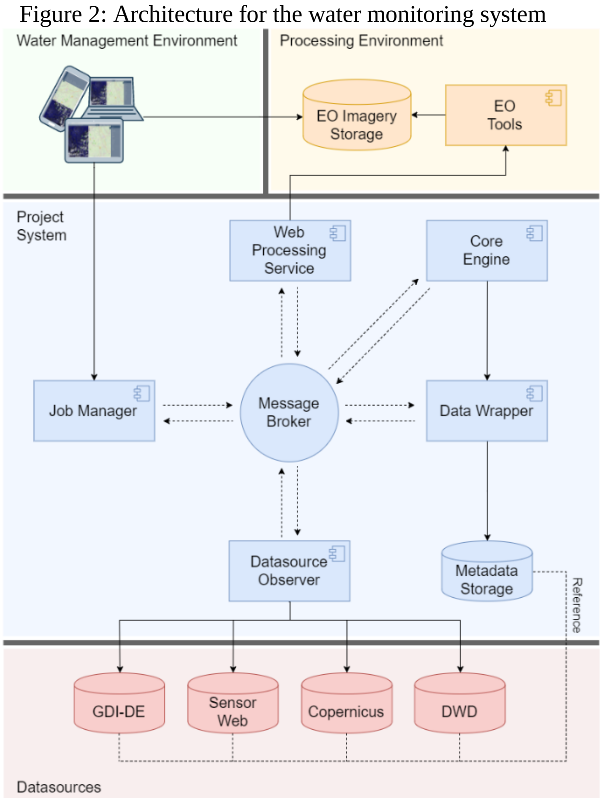


Figure 3: Example JSON message for a Job Definition


```
{
  "name": "interannual-soil-moisture-detection",
  "description": "This Job defines an interannual
                soil moisture detection.",
  "created": "2018-12-11T12:03:26.006Z",
  "lastExecution": "2018-12-11T12:03:26.006Z",
  "execution": {
    "pattern": "*/2 * * * *"
  },
  "temporalCoverage": {
    "duration": "PT1M",
    "previousExecution": false
  },
  "areaOfInterest": {
    "extent": [7.000, 52.000, 7.100, 52.100]
  },
  "processingTool": "soil-moisture-detection",
  "inputs": [
    {
      "sourceType": "CopernicusSubsetDefinition",
      "id": "LCC-INPUT-01",
      "missionId": "S2A",
      "productLevel": "MSIL1C"
    },
    {
      "id": "LCC-INPUT-02",
      "sourceType": "SensorWebSubsetDefinition",
      "serviceUrl": "sensorweb.demo.org/service",
      "offering": "ws2500",
      "procedure": "ws2500",
      "featureOfInterest": "ELV-WS2500",
      "observedProperty": "Humidity"
    }
  ]
}
```


4.2 Domain Model

The architecture of our monitoring system is based on the publish/subscribe pattern. For this we need a consistent contract for message exchanges between the single components that consider temporal and spatial aspects as well as process execution descriptions. Once the execution of a certain analysis process is planned, all components involved must be notified about the execution time and the required input data. A notification for the Observer component has to include information about required input data, it must search for. In contrast, the scheduling component requires a time definition for planning the process execution. Hence, the exchanged messages follow a domain model that is defined via the OpenAPI specification. Figure 3 depicts the key aspects of this model.

4.2.1 Process Execution Intervals

The execution time definition of an analysis process is similar to that of a crontab entry. A crontab, commonly used in UNIX environments, is a table that contains fields for time definitions and a command that is scheduled by a cron daemon at a specified time. We use the following time definition pattern based on the Open Group Base Specifications (IEEE and the Open Group, 2018):

<minute> <hour> <day-of-month> <month> <day-of-week>

Accordingly, an analysis process that is to be executed every Monday at 0:00 am will be planned as follows: *0 0 * * 1*

Our domain model currently only comprises time events but we also plan to include incident-based process executions (e.g. water level exceeds a certain threshold) in further developments.

4.2.2 Spatio-Temporal Coverage

Since hydrological models are limited to certain spatial and temporal extents, the analysis processes are as well. Our domain model considers these demands by including definitions for the spatio-temporal coverage. A process' area of interest is specified analogous to the GeoJSON bounding box as WGS84 coordinates.

In contrast to the static spatial definition, the time extent for a process depends on the execution time. Since the processes will be executed periodically, the temporal coverage of the input data must be adapted for every single execution. To fulfill this requirement, the temporal coverage of input data must be defined as backwards duration, beginning from execution time, in ISO8601 format. For instance, P1M will cover the last month before execution.

4.2.3 Input Data Subset Definitions

Each analysis process deals with specific datasets which can originate from different data sources that are accessible by certain data services. Usually, those services provide different data subsets that are referenced by unique identifiers. The system's domain model must consider these aspects, since the input data for a process are described as subset definitions. Until now, our system is designed to support Sensor Web and Copernicus data services as well as data services within a Spatial Data Infrastructure (SDI) that are indexed in a Catalog Service for the Web (CSW) (Table 1).

Table 1: Data Source Subset Identifier

Data Source	Subset Identifiers	Description
Sensor Web	Service URL	Endpoint of a SOS
	Offering	Offerings provided by the SOS
	Feature of Interest	Feature of Interest for a measurement
	Observed Property	The property that is observed by a sensor
	Procedure	The sensor
Copernicus	Mission ID	Identifier of the Sentinel mission
	Product Level	The Sentinel product level
Catalog Service	Service URL	Endpoint of a CSW
	Dataset Identifier	Identifier of the CSW dataset

These identifiers do not reference a specific dataset, but rather define a subset of all the available datasets provided by a service. Future plans comprise the possibility to consider additional data sources, such as meteorological data provided by weather services.

4.3 Event-driven Processing

An efficient platform for processing Earth Observation data needs to be able to react to the availability of new data immediately. As a result, the system architecture is designed around the publish/subscribe message exchange pattern.

Dedicated components within the architecture observe the different data sources depending on the Jobs registered in the overall system. Jobs define the required inputs for processing by providing a set of subsetting parameters (e.g. area of interest for Sentinel imagery, measurement stations for Sensor Web data). Thus, these components are able to observe the data centers efficiently. If Sentinel input is required, a component regularly searches the Copernicus Open Access Hub API for new imagery for the given area of interest with sufficient quality (i.e. cloud coverage and spatial overlap). If new data candidates are identified, a metadata reference to these is communicated to the internal message broker component. Interested components (e.g. pending Job executions) subscribe to the data and can immediately start the processing once the desired data is available. Figure 3 illustrates the workflow.

An important aspect is the provision of metadata (which contain the reference to the data) instead of the actual data. A WPS component is responsible for resolving the reference and retrieving the actual data in order to execute the tools by providing the data. This pattern facilitates efficient processing, i.e. processing near the data's physical storage location.

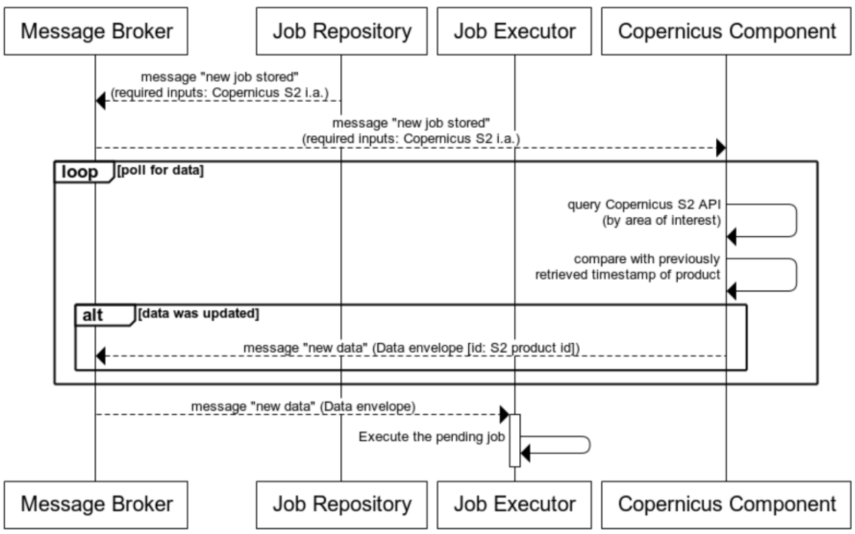
5 Future Work

In our paper, we presented an architecture that supports automatic workflows for processing heterogenous in-situ and remote sensing data as well as data provided by services that are associated with a certain SDI in order to provide innovative analytics services for an optimized water monitoring. We described different approaches that refer to the various challenges, we have to deal with when designing an event-driven system that is ready to exploit existing infrastructures for processing Copernicus data. The microservice-oriented design of our system components enables the system's deployment on platforms that provide container virtualization environments, such as DIAS or CODE-DE. In addition, the publish/subscribe pattern supports the triggering of processing workflows for occurring events, such as the availability of certain required datasets.

Since the concept of this paper is work in progress, some challenges remain and will be addressed in the future. We are planning to encapsulate Earth Observation tools via the WPS interface as we described in the architecture overview section. As a result, we must deal with the different APIs those tools provide. An easy way to use the Sentinel Application Platform with the Sentinel Toolboxes would be to integrate the required Sentinel Toolbox modules via a Java API within the WPS process implementation. In contrast, the Orfeo Toolbox provides a CLI or Python interface that requires a custom approach for bindings to the WPS component.

Furthermore, we have to evaluate the different Copernicus processing platforms to see whether they are able to provide a suitable environment in which our system can operate. Such a platform should enable easy access to the Sentinel datasets as well as the deployment of custom process algorithms. We are currently implementing our system within a demo cloud environment to analyze its cloud readiness. However, we plan to deploy the system in an operational cloud environment like the ones provided by some Copernicus data platforms, in order to exploit its whole potential.

Figure 3: Publish/Subscribe Workflow for Observing Data Centers.



Acknowledgement

This work has been funded by the Federal Ministry of Transport and Digital Infrastructure (Germany), BMVI, as part of the mFund program.

References

Bigagli, L., and Rieke, M. (2017). The new OGC Publish/Subscribe Standard-applications in the Sensor Web and the Aviation domain. In: *Open Geospatial Data, Software and Standards*, 2(18).

Botts, M., Percivall, G., Reed, C., and Davidson, J. (2006). OGC Sensor Web Enablement: Overview and High Level Architecture. In: *International conference on GeoSensor Networks*, pp. 175-190.

European Commission (2018). *Report on the "Copernicus For Water Management Workshop"*. [Online] Available from: https://copernicuswater.eventsite.be/file/19/Copernicus_for_Water_Management_Workshop_Report/

IEEE and The Open Group (2018). *The Open Group Base Specifications Issue 7, 2018 edition - crontab*. [Online] Available from: http://pubs.opengroup.org/onlinepubs/9699919799/utilities/crontab.html [Accessed 21st January 2019].

López López, P., Sutanudjaja, E. H., Schellekens, J., Sterk, G., and Bierkens, M. F. P. (2017). Calibration of a large-scale hydrological model using satellite-based soil moisture and evapotranspiration products. In: *Hydrology and Earth System Sciences*, 21(6), 3125-3144.

Rieke, M., Bigagli, L., Herle, S., Jirka, S., Kotsev, A., Liebig, T., Malewski, C., Paschke T. and Stasch, C. (2018). Geospatial IoT-The Need for Event-Driven Architectures. In: *Contemporary Spatial Data Infrastructures, International Journal of Geo-Information*, 7(10).

Seibert, J. and Beven, K. J. (2009). Gauging the ungauged basin: how many discharge measurements are needed? In: *Hydrology and Earth System Sciences*, 13(6), 883-892.

Stasch, C., Pross, B., Gräler, B., Malewski, C., Förster, C., and Jirka, S. (2018). Coupling sensor observation services and web processing services for online geoprocessing in water dam monitoring. In: *International Journal of Digital Earth*, 11(1), 64-78.

Storch, T., Reck, C., Holzwarth, S., and Keuck, V. (2018). Code-De – the German Operational Environment for Accessing and Processing Copernicus Sentinel Products. In: *Proceedings of IGARSS 2018*, pp. 6520-6523.