LOOSE

Technologies for the Management
of Long EO Data Time Series

Executive Summary (EXE)

J. Meyer-Arnek (Editor), S. Achtsnit, T. Adams, B. Buckl, V. Craciunescu, M. Eichhorn, T. Heinen, S. Meißl, M. Metz, M. Neagul, J. Ungar, J. Zeidler

Table of Contents

Revision: 1.0

Submission Date: 2022-09-16

Document Identifier: LOOSE-DLR-EXE-4101

Authors: J. Meyer-Arnek (Editor), S. Achtsnit, T. Adams, B. Buckl, V. Craciunescu, M. Eichhorn, T. Heinen, S. Meißl, M. Metz, M. Neagul, J. Ungar, J. Zeidler

Reference to ESA invitation to tender: ESA/AO/1-9581/19/I-DT

Reference to DLR Project: 3023835

Chapter 1. Introduction

The continuously increasing amount of long-term and of historic data in EO facilities in the form of online datasets and archives makes it necessary to address technologies for the longterm management of these data sets, including their consolidation, preservation, and continuation across multiple missions. The management of long EO data time series of continuing or historic missions, with more than 20 years of data available already today, requires technical solutions and technologies which differ considerably from the ones exploited by existing systems.

The ESA project LOOSE (Technologies for the Management of LOng EO Data Time Series) enables investigating, testing and implementing new technologies on long time series. This includes ingestion, discovery, exploitation optimized access, processing and optmized analysis of EO data timeseries. LOOSE partners are DLR (Oberpfaffenhofen), EOX (Vienna), Terrasigna (Bucharest) and Mundialis (Bonn).

For specific tasks (ingestion, discovery, access, processing, analysis) a multitude of completely different mature open source components is available. LOOSE combines functionally similar solutions from different heritages into a comprehensive framework. LOOSE even supports parallelism in a way that multiple solutions for the identical task are available and the application developer is invited to chose between these different components during implementation (e. g. "Geoserver" versus "EOXServer").

Result is a "blueprint architecture concept" which focuses on the interfaces between components and takes innovative concepts such as Data Analysis Processing API and Data Cubes offering Discrete Global Grid Systems into consideration.

Most important is that the validity of the LOOSE blueprint architecture is demonstrated in three different real-world application pilots. These applications are covering totally different thematic areas: Agricultural monitoring, monitoring urbanization globally and supporting fishery in the Black Sea. In more details they are described in section \ref*{sec:pilots}.

1.1. Purpose and Scope

The LOOSE project develops and evaluates technologies and solutions for interoperable processing and timeseries analysis in online platforms derived from contemporary use cases and stakeholder needs.

LOOSE aims at building up an innovative architecture for efficient timeseries processing and analysis.

The executive summary concentrates on the evaluation of current and state-of-the-art technologies with respect to their specific applicability. Each section discusses usage of technologies

- from the user perspective (what is the benefit for the user) as well as
- from the developer perspective - specifically with the question: Will the developer’s effort be reduced when deploying/configuring/implementing a certain technology.

In particular, pros and cons of the analysed and/or implemented technologies are closely investigated.

In addition, the resulting LOOSE architecture is applied and evaluated in three "real world scenarios" (=pilot applications): Technologies which have been analysed during the project are applied by the pilots and are evaluated.

1.2. Structure of the Document

The remainder of the document is organized as follows.

Section 2 - Architecture

Explains basic principles of the underlying architecture for handling and processing EO data timeseries

Section 3 - Ingestion

Gives an overview of the components used and developed for data ingestion into the *LOOSE prototype*.

Section 4 - Discovery

Gives an overview of the standards evaluated for data data discovery within the *LOOSE prototype*.

Section 5 - Processing

Gives an overview of the technologies evaluated for data processing within the *LOOSE prototype*.

Section 6 - Pilot applications

Gives an overview of the real world pilot applications which were used to evaluate the technologies.

Section 7 - Open Source Contributions

Gives an overview of the Open Source libraries and components released during the *LOOSE* project.

Section 8 - Conclusion

Provides an overview on the open source components developed during *LOOSE*.

1.3. Reference Documents

Reference	Description	Version
[LOOSE Statement of Work]	LOOSE ESA-EOPG-EOPGM-SOW-3	Issue 1, Rev. 0
[LOOSE-UC-TN]	LOOSE-DLR-TN-1100: Use Cases Technical Note https://teamsites-extranet.dlr.de/dfd/loose/90_DeliveryPackages/2020-05-UR/LOOSE-DLR-TN-1100_UseCasesTechnicalNote_1.1.pdf	Issue 1.1, 07/08/2020
[LOOSE-SDD]	LOOSE-DLR-SDD-2102: System Design Document https://teamsites-extranet.dlr.de/dfd/loose/90_DeliveryPackages/2021-08-PM-2/LOOSE-DLR-SDD-2102_SoftwareDesignDocument_v1.1.pdf	Issue 1.1, 20/10/2021
[LOOSE-TA-TN]	LOOSE-DLR-TN-1200: Technical Analysis Technical Note https://teamsites-extranet.dlr.de/dfd/loose/90_DeliveryPackages/2020-05-UR/LOOSE-EOX-TN-1200_TechnologyAnalysisTechnicalNote_1.1.pdf	Issue 1.1, 17/06/2020
[LOOSE-SSS]	LOOSE-TER-SSS-1301: Software System Specification https://teamsites-extranet.dlr.de/dfd/loose/90_DeliveryPackages/2020-05-UR/LOOSE-TER-SSS-1301_SoftwareSystemSpecification_1.1.pdf	Issue 1.1, 29/05/2020
[LOOSE-EDRTN]	LOOSE-DLR-EDR-3401: Evaluation and Feedback Analysis https://esa.pages.eox.at/loose/docs/LOOSE-DLR-EDR-3401_UserFeedback_1.0/LOOSE-DLR-EDR-3401_UserFeedback.html	Issue 1.0, 13/09/2022

Reference	Description	Version
[EOEPCA-SSD]	EOEPCA.SDD.001: Master System Design Document https://eoepca.github.io/master-system-design/published/v1.0/	Issue 1.0, 02/08/2019
[ECSS-E-ST-40C]	Space Engineering – Software	06/03/2009
[GEO-OSEO-REST]	GeoServer OpenSearch for EO: Automation with the administration REST API https://docs.geoserver.org/latest/en/user/community/opensearch-eo/automation.html	accessed 14/10/2020
[STAC-SPEC]	SpatioTemporal Asset Catalog (STAC) specification https://github.com/radiantearth/stac-spec	v1.0.0-beta.2 - accessed 14/10/2020
[STAC-API-SPEC]	SpatioTemporal Asset Catalog (STAC) API specification https://github.com/radiantearth/stac-api-spec	accessed 14/10/2020
[OGC-13-026r8]	OGC OpenSearch Extension for Earth Observation https://docs.opengeospatial.org/is/13-026r8/13-026r8.html	OGC 13-026r8, 06/07/2016
[OGC 17-047]	OGC OpenSearch-EO GeoJSON(-LD) Response Encoding Standard https://docs.opengeospatial.org/is/17-047r1/17-047r1.html	OGC 17-047, 04/27/2020

Chapter 2. Architecture

The LOOSE prototype is a self-contained system and assumes that several external systems (grey) are available. First and foremost, a *Platform* is required that provide the runtime environment in which the system services are deployed. The software components providing these services need to be accessible in external *Repositories*. A connection to the *Archive* holding the historic EO data is needed to demonstrate bulk reload capabilities.

Figure 1 shows the overall design of the *LOOSE Prototype*.

The prototype consists of several system *Services* (blue) mainly providing access to EO resources (see [LOOSE-SDD]) and domain-specific *Applications* exploiting and demonstrating the capabilities of the services.

The services are implemented with different *Software Components* (green, see [LOOSE-SDD]) that are defined by each of the *Applications* individually, depending of it's needs and requirements. Furthermore service-specific data repositories are managed by the each service separately. Each service provides a set of *Interfaces* that are used by either internal (purple) and external (yellow) components, systems or applications.

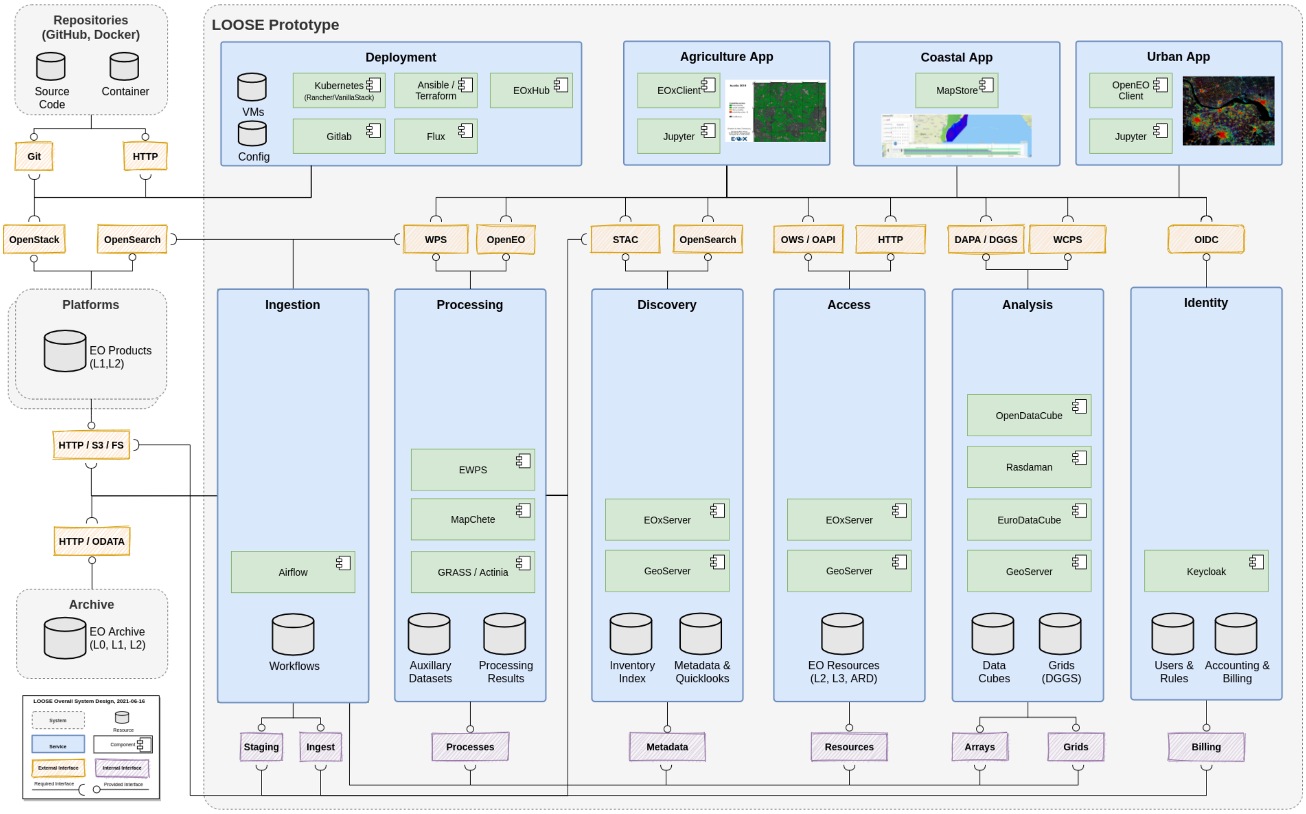

Figure 1. Overall design of the LOOSE Prototype

Services are considered as high-level components exposing interfaces. The service interfaces are categorized as follows:

- Internal (violet): interfaces not exposed to components and users outside of the system
- External (yellow): interfaces accessible by external systems and applications
- Required: interfaces that are needed for nominal operation

Chapter 3. Ingestion

In order to enable EO data processing, these data need to be available or accessible on the *LOOSE* platform. Automated workflows ensure that EO data are properly ingested and registered into the LOOSE architecture. These workflows either need to be scheduled for periodic executions (e. g. once per day) or "on demand" when data becomes available on a pick-up-point. The requirement for automated EO data ingestion accounts

- for EO data from archives that needs to become accessible within our architecture as well as
- for products being generated by dedicated processors.

Overall objective is to make EO data easily discoverable and accessible for any kind of application.

3.1. Development

In the framework of LOOSE, Apache Airflow was applied for this task. Apache Airflow uses the concept of workflows: Each task — such as a collection-specific EO data ingestion — is considered as an ETL-process (= Extract, Transform, Load). These processes are implemented as "Directed Acyclic Graphs" (DAGs).

A number of EO data ingestion workflows (DAGs) have been developed for the ingestion of EO products required on the *LOOSE* platform. For dedicated purposes, data from Sentinel 2 and Sentinel 5P have been ingested into the platform.

Airflow’s capability of single EO product ingestion has been extended with another function: Support of bulk reload from Long-Term-Archives such as the DLR Long Term Archive (LTA).

In the LOOSE project, a Python-library was developed to utilize the LTA-interface as specified by ESA (see ???:ESA-EOPG-EOPGC-IF-2) in the framework of the Copernicus Space Component (CSC) Ground Segment (GS).

This represents the "bulk reload scenario" (which was outlined as a scenario in the Use-Case Technical Note, UC-TN-document, https://teamsites-extranet.dlr.de/dfd/loose/90_DeliveryPackages/2021-01-MTR/LOOSE-DLR-TN-1100_UseCasesTechnicalNote_1.2.pdf):

- single products or
- bulks of EO products filtered according to spatio-temporal criteria

can easily be ingested into the *LOOSE* platform.

Since data download from an offline tape archive (such as the LTA) likely results in high latencies, the request becomes asynchroneous. And this means that the request handling becomes very complicated.

The Python library completely performs the complex multi-step asynchroneous communication between the client (the *LOOSE* platform) and the LTA. In Apache Airflow, the developer can also trigger additional workflows which are performing follow-up-actions like dedicated processing or ingestion.

The library is open-source (GNU GENERAL PUBLIC LICENSE Version 3) and is available here: https://github.com/dlr-eoc/aip-client.

3.2. Conclusion

For a developer who is implementing workflows for data ingestion or processing, the Apache Airflow is a flexible and easily usable tool. The Apache Airflow web frontend provides overviews

To perform bulk reload from Long-Term-Archives (LTA), the "aip-client"-library (accessible at https://github.com/dlr-eoc/aip-client and published under GNU GENERAL PUBLIC LICENSE Version 3) was developed in the LOOSE project. This single library completely covers the complex communication caused by asynchroneous processes. Using this library, bulk reload becomes straight forward.

Chapter 4. Discovery

Data discovery of collections and products is a central part for data access, data visualization, data processing, and data analysis. All of these components need to know about available EO data as well as processing results. Thus, the discovery component needs to be connected closely to the other components and standardized interfaces, which are compliant to these components, are needed.

The discovery component of LOOSE describes how EO collections, EO products, user-generated processing results as well as available applications and processing services are identified for visualization, access and processing.

LOOSE evaluates and implements two EO metadata specifications for data discovery:

- OpenSearch for EO [OGC-13-026r8]
- STAC [STAC-SPEC]

OpenSearch EO became "the" state-of-the-art interface for EO-Product discovery. The current OpenSearch-implementation in GeoServer was formerly supported by the predecessor project EVO-ODAS. OpenSearch is implemented in GeoServer (https://docs.geoserver.org/main/en/user/community/opensearch-eo/intro.html) as well as EOxServer (see https://docs.eoxserver.org/en/stable/users/services/opensearch.html).

However, when seen from today’s perspective, OpenSearch yields some disadvantages:

- OpenSearch-results are returned as Extensible Markup Language (XML) which usually needs to be translated to JSON within most of the clients: This reduces interoperability for state-of-the-art applications
- In order to discover a dataset, the "two-step-search" is required: In the first search the collection is discovered, the required product in the second search step.

4.1. Development

The LOOSE design include a central, project specific catalogue for EO collections and products that is being filled by the data workflows of the Ingestion Service (see Figure 1).

In the course of the project, STAC support [STAC-SPEC] has been added to all relevant components in LOOSE, namely GeoServer, GRASS GIS / Actinia and EWPS.

In GeoServer, metadata for EO collections, EO products, and processing results is stored in one single a Postgres database. For data discovery, both specifications STAC API [STAC-

API-SPEC] and OGC OpenSearch for EO [OGC-13-026r8] are served within LOOSE from this Postgres database. Data curators have the opportunity to add additional metadata tags which mainly appear in STAC metadata (extending the standard), but are not required to take any action in adjusting metadata holdings.

4.2. Conclusion

- All relevant *LOOSE prototype*-components now support the STAC specifications [STAC-SPEC] (besides OpenSeach EO [OGC-13-026r8]).
- All required implementation work was enabled by the LOOSE project.
- All resulting software components are released as open source.
- From the developer’s perspective: STAC returns its items in JSON format, which can be considered as native format for most state-of-the-art applications. No parsing of XML is required in the applications.
- From the data curator’s perspective: By GeoServer OpenSearch and STAC are served from one identical Postgres-database instance, there is no need to adjust or duplicate metadata holdings.

Chapter 5. Processing

The Processing Service of *LOOSE* provides functionality for processing geospatial data using platform provided or user defined processors. The relevant interfaces and services for processing are shown in Figure 1, relevant components which have been investigated are:

- openEO / Acinia / GRASS GIS,
- X-array / X-array-spatial + dask,
- Data Analysis and Processing API (DAPA),
- EWPS and
- MapChete Hub.

5.1. Development

In the course of the LOOSE project, the above mentioned components were

- evaluated,
- improved and
- additional functionalities included.

Usage of innovative APIs such as

- openEO,
- DAPA,
- OGC API Processes,
- STAC

have also been investigated and integrated.

The evaluation was mainly conducted in the framework of the "real-world"-pilot applications (see next section).

5.2. Conclusion

Major improvements for the processing components were:

- including STAC metadata support (openEO/Actinia/GRASS GIS and EWPS),

- adding additional geospatial processes to existing components (Actina + GRASS GIS),
- apply standardized interfaces for communication within the processing environment (MapChete Hub) and
- develop a transparent computing environment resilient to errors and efficient workload distribution based on dask and Kubernetes.

Chapter 6. Pilot Applications

Three "real-world" scenarios are described in the [LOOSE-UC-TN]. By implementing these "real-world"-scenarios, the components and interfaces discussed in the previous sections are evaluated for their fitness.

6.1. Urban Application Pilot

A continuous and reliable monitoring of urbanization is of key importance to accurately estimate the distribution of the expanding human population, and to evaluate its effects on the use of resources, infrastructure needs, socioeconomic development, etc. In recent years, the increased availability of EO data, along with the development of sophisticated machine learning algorithms has facilitated this task, by allowing the production of geospatial datasets that describe the extent and distribution of human settlements at local, regional and global scales.

WSF is routinely used by the World Bank, Asian Development Bank, UN-HABITAT, International Committee of the Red Cross as well as by numerous research institutions as a basis for their own research.

WSF data is open and freely available to the public via the DLR/EOC Geoservice.

In the LOOSE project, the processing of the WSF 2019 with openEO / Actinia / GRASS GIS was intensely evaluated. All components of this processing chain were extended with new functionalities and processes. All required index-like EO-pre-products were successfully generated by the planned approach using openEO. The openEO process graph finally contained about 160 single processing steps.

However, the openEO specification does not allow handling classification information (vector type): So the last step could not be implemented due to limitations of openEO.

The processing chain openEO / Actinia and GRASS GIS was running smoothly and stable on the LOOSE Kubernetes cluster.

6.2. Marine Application Pilot

To foster Terrasigna’s stakeholder - the Romania’s National Institute for Marine Research and Development "Grigore Antipa" (NIMRD) - in implementing the “Blue Growth” strategy, LOOSE technology will be applied. This strategy also offers significant support to other initiatives targeting a healthy, resilient and sustainable marine basin. On the European level, this strategy for the Black Sea is backed by the newly launched Strategic Research and

Innovation Agenda and the Common Maritime Agenda for the Black Sea.

In the LOOSE project, the EWPS processing component was further developed:

- Support for OGC API Processes (REST API)
- Support of Kubernetes functionalities: Jobs are run as Kubernetes jobs
- Jobs are triggered with input data provided directly by reference to a data store (Shared Filesystem Location, HTTP Endpoint – S3)
- Jobs are triggered with input data encapsulated in STAC Catalogues

6.3. Agricultural Application Pilot

In the context of the European Common Agricultural Policy (CAP), related subsidy claims require that certain agricultural practices can be monitored and verified, e.g. the presence/absence of mowing in grassland, or the occurrence of ploughing during a specific seasonal time window or the rapid growth of vegetative cover during certain time period.

The Agri App is a tool to help paying agencies visualize and verify farmers declarations of land usage. Farmers declare and apply for subsidies after which paying agencies have to verify submissions and provide the payments. Paying agencies use the Agricultural App to evaluate these declarations.

A processing & analytic workflow performs claim eligibility checks for national/sub-national LPIS (Land Parcel Identification System) parcels. Eligibility checks shall perform an automated evaluation of subsidy claims and ascertain if certain land management process has occurred, e.g. ploughing, crop development, harvesting, catch crop cover, all with respect to their country and crop/claim type specific reference period.

In the LOOSE project, Mapchete Hub - which is an asynchronous processing service around the processing engine mapchete - was significantly improved. An API for communication between clients and the mapchete Hub was designed in alignment with the OGC API Processes. Dask is used to achieve workload-distribution to the nodes. Together with the underlying Kubernetes, a resilient compute system is created.

Chapter 7. Contributions to Open Source

During the course of the project, numerous software improvements were made available to the open source community.

- GeoServer (with support of GeoSolutions):
  - STAC Service within OSEO Plugin (completed)
  - STAC Data Store (ongoing)
  - OGC Feature Templating Plugin (completed)
  - OGC Coverage API Improvements (ongoing)
  - DGGS / DAPA Plugin (testing)
- STAC Spec
  - EO Extension – eo:snow_cover definition
  - Tiled Assets Extension: https://github.com/stac-extensions/tiled-assets
    - GDAL driver (implemented by 3rd party): https://gdal.org/drivers/raster/stacta.html
- PySTAC
  - Core – mediatype support for items selection
  - CLI – and mediatype support
- dask
  - fix for disk I/O counters: https://github.com/dask/distributed/pull/6093
- EOxServer
  - S3 support: https://github.com/EOxServer/eoxserver/pull/461
- mapchete
  - various improvements (dask, STAC writer, …): https://github.com/ungarj/mapchete
- Long-Term-Archive Bulk Reload:
  - Library for handling bulk reloads: https://github.com/dlr-eoc/aip-client

Chapter 8. Conclusion

The ESA GSTP project LOOSE develops and evaluates technologies and solutions for interoperable processing and timeseries analysis in online platforms derived from contemporary use cases and stakeholder needs.

LOOSE aims at building up an innovative architecture for efficient timeseries processing and analysis.

This project concentrates on the evaluation of current and state-of-the-art technologies with respect to their specific applicability. Each section discusses usage of technologies

- from the user perspective (what is the benefit for the user) as well as
- from the developer perspective - specifically with the question: Will the developer’s effort be reduced when deploying/configuring/implementing a certain technology.

Significant improvements have been achieved by

- integrating additional EO data processes into existing solutions (e. g. Actinia and GRASS GIS),
- ensuring a better resiliance to availability of compute resources within a compute environment and better workload charing amoung workers on the system by using dask and Kubernetes by adjusting components,
- integrating interoperability with new standards (STAC) into existing solutions and
- combining existing standards in innovative ways (e. g. integration of Common Workflow Language into EWPS)-

All these contributions improved the applicability of the evaluated and discussed components from the developers perspective.

All these improvements were enabled by building upon a stable Kubernetes compute environment. The required compute resources were provided "terra_Byte", a joint activity of DLR and Leibniz Supercomputing Centre in Garching/Germany.

All improvements are already (or will be) released as open source software under various open licenses.