The Euclid Science Ground Segment distributed infrastructure: system integration and challenges

Marco Frailis[1], Andrey Belikov[2], Kevin Benson[3], Andrea Bonchi[4], Christophe Dabin[5], Anne Ealet[6], Marco Fumana[7], Catherine Grenet[8], Mark Holliman[9], Gianmarco Maggio[1], Davide Maino[10], Henry J. McCracken[8], Martin Melchior[11], Antonello Piemonte[12], Gianluca Polenta[4], Maurice Poncet[5], Paolo Luigi Scala[7], Santiago Serrano[13], O. Rees Williams[2]

(1) INAF - Osservatorio Astronomico di Trieste, Italy, (2) University of Groningen, Netherlands, (3) Mullard Space Science Laboratory / UCL, United Kingdom, (4) ASDC/INAF - Osservatorio Astronomico di Roma, Italy, (5) CNES Toulouse Centre National d'Etudes Spatiales, France, (6) LAM - Laboratoire d'Astrophysique de Marseille, France (7) INAF - IASF Milano, Italy, (8) CNRS - Institut d'Astrophysique de Paris, France, (9) University of Edinburgh, United Kingdom, (10) University of Milano, Italy, (11) University of Applied Science of Northwestern Switzerland, Switzerland, (12) Max Planck Institute for Extraterrestrial Physics, Garching, Germany, (13) IEEC - Institut d'Estudis Espacials de Catalunya, Barcellona, Spain

Contact: frailis@oats.inaf.it

***Abstract*:**

The Science Ground Segment (SGS) of the Euclid mission provides distributed and redundant data storage and processing, federating nine Science Data Centres (SDCs) and a Science Operations Centre.

The SGS reference architecture is based on loosely coupled systems and services, broadly organized into a common infrastructure of transverse software components and the scientific data Processing Functions. The SGS common infrastructure includes:

1) the Euclid Archive System (EAS), a central metadata repository which inventories, indexes and localizes the huge amount of distributed data;

2) a Distributed Storage System, providing a unified view of the SDCs storage systems and supporting several transfer protocols;

3) an Infrastructure Abstraction Layer, isolating the scientific data processing software from the underlying IT infrastructure and providing a common, lightweight workflow management system;

4) a Cmmmon Orchestration System, performing a balanced distribution of data and processing among the SDCs.

The Euclid scientific data processing levels are decomposed into eleven Processing Functions (PFs), which are the highest-level break-down of the complete processing. They are developed by distributed teams, with the constraint that each PF pipeline should run in any SDC.

Virtualization is another key element of the SGS infrastructure. The EuclidVM is a lightweight virtual machine, deployed in any SDC processing node, with a reference operating system, selected stable software libraries and "dynamic" installation of the Euclid PFs based on the CernVM-FS file system.

We present the status of the Euclid SGS software infrastructure, the prototypes developed and the continuous system integration and testing performed through the Euclid "SGS Challenges".

The Euclid Science Ground Segment

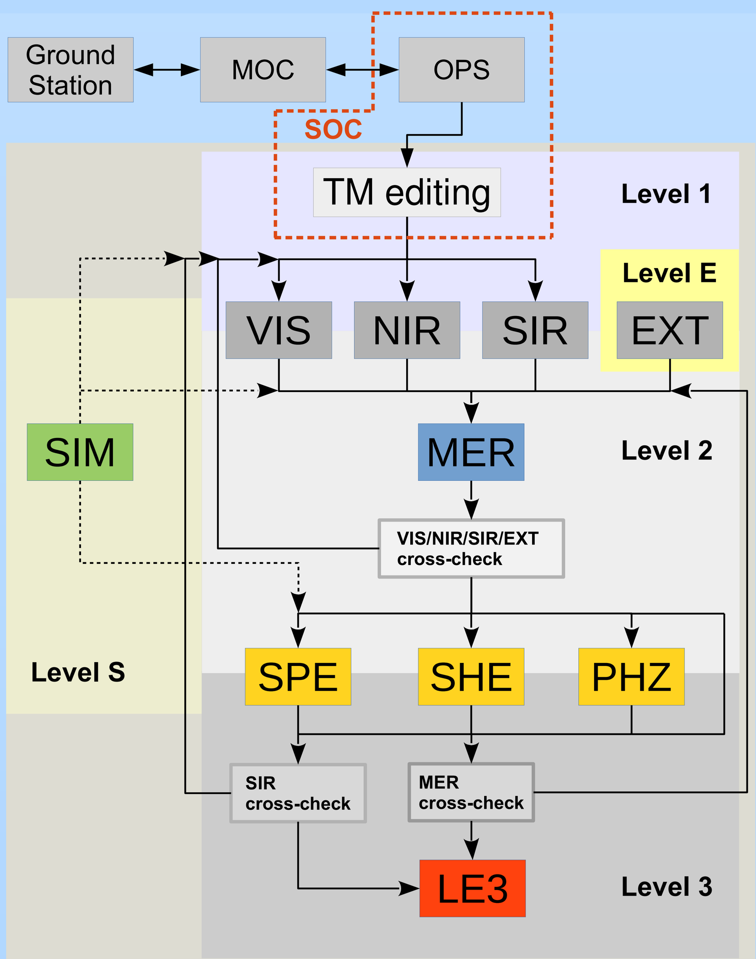
The Euclid SGS is composed by the ESA Science Operation Center (SOC) and the Euclid Consortium SGS. The SOC is in charge of the mission planning, of the first consistency and quality checks and of the production of quick-look-quality data for public distribution. The Euclid Consortium SGS, composed by eight European SDCs and one USA SDC, is in charge of the instrument-related processing, production of science data products, simulations, ingestion of external data and, in general, of designing, developing, integrating and operating the Scientific data processing.

The Euclid mission will generate a large amount of data: heavy simulations will be needed before flight operations, and several re-processing - from raw data up to the science products - will multiply the data volume by dozens. In addition, a large amount of external data will be gathered from ground-based observations. A full Euclid data release will generate 26 Pbytes of data (including external data), reaching a grand total of 175 Pbytes.

The Euclid Processing Functions

The different data processing levels of the euclid SGS are connected with logical data processing functions. These processing functions have been defined by considering that they represent self-contained processing units, i.e. they represent they highest-level break-down of the complete pipeline that can be achieved with units that communicate only with the help of a central metadata and data repository (the Euclid Archive System). The picture below provides an overview of the SGS processing functions:

- VIS, NIR, EXT: production of fully calibrated photometric exposures from Euclid and ground-based surveys (VIS: visible imaging; NIR: near infrared imaging; EXT: external data)
- SIR: production of fully calibrated 1D spectra extracted from the NISP spectroscopic exposures
- MER: production of a source catalog containing consistent photometric and spectroscopic measurements
- PHZ: production of the photometric redshift for all catalogued sources
- SPE: production of spectroscopic redshifts for all sources with spectra
- SHE: measurements of galaxy shapes
- LE3: production of all high-level science products
- SIM: production of all the simulated data necessary to validate the data processing stages, and to calibrate observational or method biases



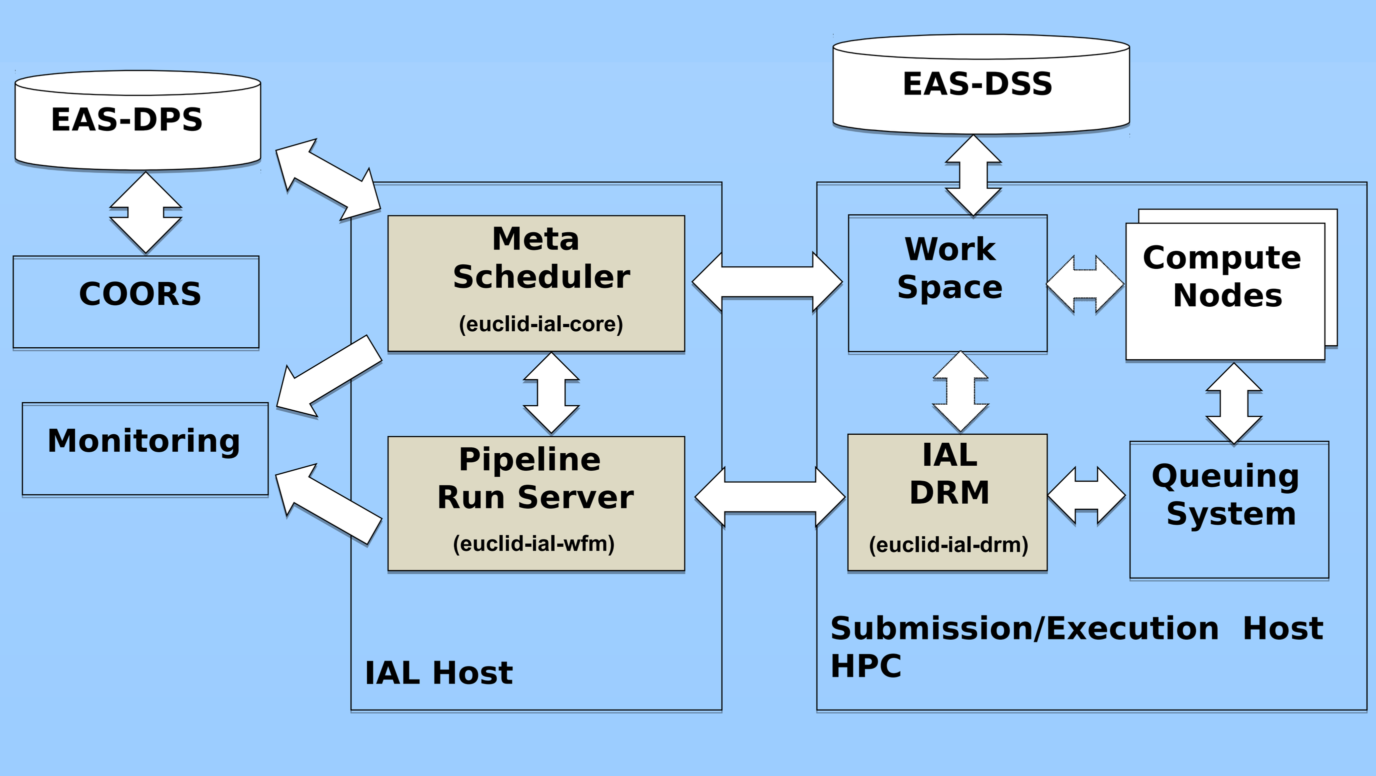
The SGS common infrastructure

The large amount of data generated during the Euclid mission, together with the multiple SDC organizations involved in the Euclid Consortium, leads to a distributed processing of the Euclid data, achieved through the following architecture key principles:

- Each SGS pipeline should run on any SDC
- Each SDC is both a processing and a storage "node"
- Separation of metadata (inventory) from data (storage)
- For lower processing levels, adoption of a "map/reduce" approach, where a pipeline is designed to process an independent quantum of locally available data

Following these principles, the SGS System Engineering team has defined a common infrastructure of loosely coupled software components for the development and running of the Euclid processing functions:

- A Common Data Model (CDM), providing a central repository where all SGS components interfaces and data structures are formalized in the XSD language
- The EAS Data Processing Sytem (DPS), implementing the CDM metadata repository in a (relational) database management system and providing an abstract metadata access layer independent of the database implementation
- The EAS Distributed Storage System (DSS), supporting the copy, retrieval and movement of data files between SDCs. It is compatible with different storage solutions, including a GRID storage, an SFTP server or an iRODS data system
- A COmmon Orchestration System (COORS) for the preparation, dispatch and monitoring of pipeline processing orders and performing a load balancing depending on the Euclid scanning strategy and the resources available in each SDC
- A Monitoring and Control system (M&C), tracking availability of computing resources and the Euclid data processing performance and progress



- The Infrastructure Abstraction Layer (IAL), which makes processing functions software isolated from the Euclid Archive, the SDCs computing infrastructure and the data storage and transfer services. The IAL has three main components: a Meta Scheduler system, retrieving pipeline processing orders from the EAS and assuring that all needed pipeline inputs become locally available; a WorkFlow Manager (WFM), configuring and submitting jobs according to pipeline workflow definitions; a Distributed Resource Manager (DRM), providing an abstraction for the different batch queuing systems used by SDCs

In order to run each pipeline in any SDC and have reproducible results, another key element of the SGS infrastructure is the adoption of a reference Operating System with a set of reference software libraries and versions. Such reference environment, called EuclidVM, is shared by all SDCs and deployed as a virtual node of the computing infrastructure using either virtualization technologies (e.g. KVM) or Linux Containers (e.g. Docker). The EuclidVM is kept as a lightweight system with the adoption of the CernVM-FS, a read-only network file system shared by all SDCs to distribute and update the SGS reference environment and processing functions.

A processing function workflow

The IAL workflow engine, using the Pydron system (https://github.com/pydron/pydron), translates regular python code into an intermediated data flow graph representation. Then, it evaluates in which order to execute each task of the workflow and which of them can run in parallel.

An example of pipeline workflow definition with the IAL is shown below. It represents the NIR proto-pipeline recently integrated in the SGS infrastructure. The current NIR prototype requires in input four dither exposures of a given filter of the NISP-P instrument, the Mission Database (which provides information of the current instrument model), a master dark image separately obtained from a set of raw dark exposures, and a reference catalog to compute the astrometry.

The calibrateDither function groups a number of NIR pre-calibration steps performed on each dither. It is called four times in the main pipeline function, labelled with the @pipeline decorator. The IAL workflow engine deduces that each calibrateDither call is independent, and hence it executes them in four parallel jobs, submitted to the SDC computing infrastructure by the DRM component of the IAL.

The subsequent pipeline steps (astrometry, resampling, stacking, source detection and photometry) are performed sequentially. The outputs of the pipeline are four calibrated exposures, a stacked image and the extracted source catalog. As soon as the last workflow task completes, the IAL retrieves the outputs and ingest them automatically in the EAS, together with their metadata.

```
from euclidwf.framework.workflow_dsl import pipeline
from nir_package import runInitialize, badPixMasking, darkSubtract, crRejection, backSubtract
from nir_package import doAstrom, doResampl, doStack, catalogDetect, catalogPhot

def calibrateDither(ditherExposure, mdb, masterDark):
  initializedExposure = runInitialize(infile=ditherExposure)

  badMaskedExposure = badPixMasking(infile=initializedExposure,
                                    mdbfile=mdb)

  darkSubExposure = darkSubtract(infile=badMaskedExposure,
                                 darkcalfile=masterDark)

  crRemovedExposure = crRejection(image=darkSubExposure)

  bgRemovedExposure = backSubtract(input=crRemovedExposure)

  return bgRemovedExposure

@pipeline(('calDither1', 'calDither2', 'calDither3', 'calDither4', 'stackedImage',  ' sourceCatalog'))
def nir_pipeline(dither1, dither2, dither3, dither4, mdb, masterDark, catalog):

   ditherExposure1 = calibrateDither(ditherExposure=dither1, mdb=mdb, masterDark=masterDark)
   ditherExposure2 = calibrateDither(ditherExposure=dither2, mdb=mdb, masterDark=masterDark)
   ditherExposure3 = calibrateDither(ditherExposure=dither3, mdb=mdb, masterDark=masterDark)
   ditherExposure4 = calibrateDither(ditherExposure=dither4, mdb=mdb, masterDark=masterDark)

   calDither1,calDither2,calDither3,calDither4 = doAstrom(dither1=ditherExposure1,dither2=ditherExposure2,
                                                        dither3=ditherExposure3,dither4=ditherExposure4,
                                                        refcat_xml=catalog)

   resampDither1, resampDither2, resampDither3, resampDither4 = doResampl(caldither1=calDither1,
                                                                         caldither2=calDither2,
                                                                         caldither3=calDither3,
                                                                         caldither4=calDither4)

   stackedImage = doStack(resampledmef1=resampDither1, resampledmef2=resampDither2,
                          resampledmef3=resampDither3, resampledmef4=resampDither4)

   sourceMask = catalogDetect(image=stackedImage)

   sourceCatalog = catalogPhot(image=stackedImage,source_mask=sourceMask)

   return calDither1, calDither2, calDither3, calDither4, stackedImage, sourceCatalog
```

The SGS challenges

The assembly of a complex system as the SGS infrastructure - developed by several distributed teams - requires a continuous and incremental integration and delivery process.

This process has been implemented by the SGS as a set of planned Infrastructure and Scientific Challenges, taking into account the incremental development and maturity of the involved software components. The Infrastructure Challenges are dedicated to the integration and validation of the software components that are not "science dependent" and to their deployment in each SDC environment. Each Infrastructure Challenge is followed by a Scientific Challenge, focused on the integration and validation of the SGS processing functions.

Currently, an Infrastructure Challenge 6 has been successfully performed, testing most of the SGS common infrastructure (EAS-DPS, EAS-DSS, IAL, M&C, CernVM-FS, etc.). It has been followed by the Scientific Challenge 2, that has involved the integration of SIM, VIS, NIR and SIR processing functions. In particular, SIM has produced VIS, NISP-P and NISP-S raw dither expsoures, introducing the requested features and instrumental effects. Then VIS, NIR and SIR proto-pipelines have been succesfully deployed and run in the SDCs computing infrastructure, processing the simulated data and ingesting the obtained output products in the EAS archive.