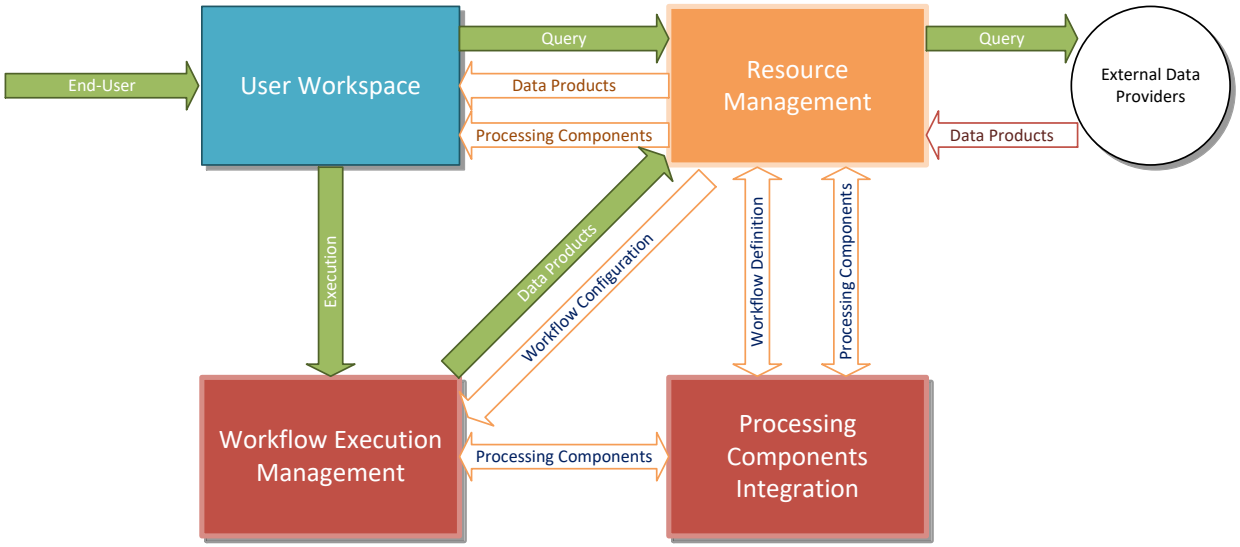
OVERVIEW

TAO (stands for Tool Augmentation by user enhancements and Orchestration) is a lightweight, generic integration and distributed orchestration framework. It allows to reuse (i.e. integrate) commonly used toolboxes (such as, but not limited to, SNAP, Orfeo Toolbox, GDAL, PolSARPro, etc.). This framework allows for processing composition and distribution in such a way that end users could define by themselves processing workflows and easily integrate additional processing modules (by processing module it is understood either a standalone executable or a script).

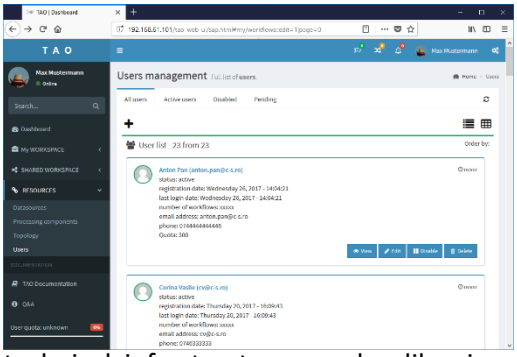
In terms of use, the TAO platform provides a mean for orchestration of heterogeneous processing components and libraries in order to process scientific data. This is achieved in following steps:

- Preparation of resources (including processing components) and data input,
- Definition of a workflow as a processing chain,
- Execution of workflows,
- Retrieval / visualization of the results.

To have a simple view of the TAO platform, the platform model is split among four main macro-components. Such a macro-component is a logical collection of components with related functions. It has no direct relationship to the software implementation.



USER MANAGEMENT



The TAO platform was designed and built to be a multi-user platform. Therefore, one of its basic features relates to the user management.

User can authenticate to the platform by several configurable mechanisms (local user base, LDAP or even single sign-on). User accounts can be linked to groups with appropriate system rights.

Since many years, EO Data and EO Data processing as well, use increasingly disk space and computing power. Even if technical infrastructures evolve likewise, computing and storage resources have a not negligible provisioning and usage cost. Therefore, we have to limit resource usage in order to offer a fair operation of the platform.

The platform, through its Authorization module, defines **quota management** by user. The quota consists in the pool of resources that may be allocated for a single user, in terms of: number of CPUs, storage space and, possibly, RAM memory amount for the execution of user workflows.

A dedicated activity monitoring service captures the required information and collects it in a database. This database with tracked processing activity and disk space consumption is used then to automatically restrict the access to resources according to the user quota.

DATA ACCESS

In general, data access refers to activities related to storing, retrieving or acting on data housed in a repository.

For TAO, the Data Access feature deals with the provisioning of data (either EO products or ancillary data) that exhibits certain criteria for workflows execution.

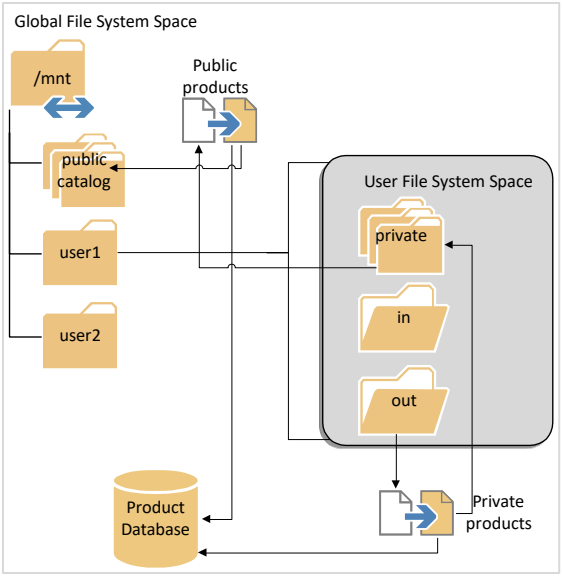
Two categories of data sources can be distinguished: local and remote.

LOCAL DATA ACCESS

A local **data product** is nothing but a descriptor for one or more files (for example, a GeoTIFF data product can be just a single raster file, while a Sentinel-2 data product is a structured collection of raster and metadata files). Consequently, a local data source is responsible with providing local data to the TAO platform to the requesting processing component(s).

Even if the local data source (comprising in the local database and file system) is not an external interface, it is important to layout its organization before describing the external interfaces of the system.

As it can be seen in the figure beside, the local data source consists of a product database, in which metadata associated with concrete data products are stored, and a local (in the platform context) file system, in which the data products are physically stored.



The product database stores basic metadata about the EO products, such as acquisition date, geographical footprint, product type, etc.

This metadata allows the users to query the local data source for products satisfying certain criteria, and is created when either products are initially imported or new products are downloaded from remote data sources.

The file system is organized such that an easier distinction can be made between public products (products that are visible/usable by all users) and private products (products visible/usable only by a specific user). This separation further allows the implementation of user quota management.

The visibility of the products is implemented at a logical level (i.e. not by physical operating system rights) in the database.

The file system structure depicted above is visible to (i.e. shared with) all the processing nodes in the TAO cluster. This is necessary in order to allow a uniform way of accessing products by processing components from remote nodes.

EXTERNAL DATA ACCESS

External data sources represent external repositories that are not under the direct control of the TAO platform (an example of such an external data source is the Copernicus Scientific Data Hub). They may implement different access interfaces and authentication/authorization schemes. This is transparent for the user of the TAO platform.

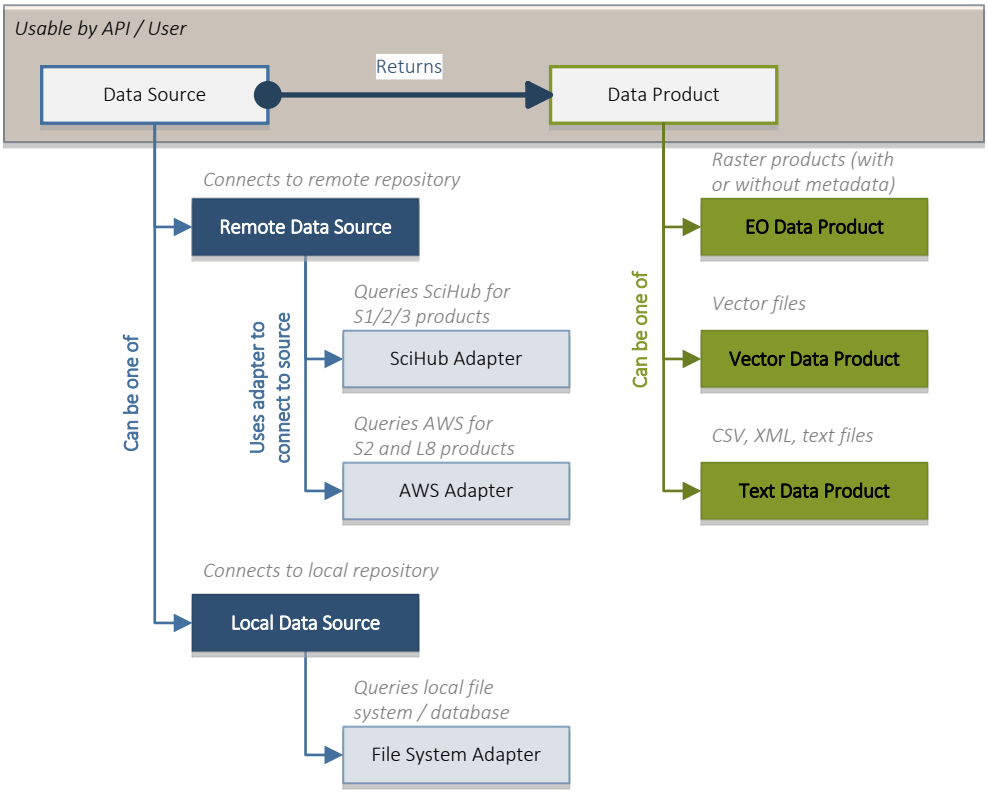
Eventually, all needed data would become local, taking also into account the defined user quotas.

Contrary to the differences that may exist between repositories access and the data formats that may be provided, several common features may be derived:

- All data can be expressed in binary form;
- There are common operations (actions) that can be performed on any data source:
    - Authenticate/authorize the user connection to the data source;
    - Query the data source for data satisfying several criteria;
    - Retrieve the data in a binary form that can be interpreted by TAO.

The framework is capable to abstract the data format and to expose them in a unitary way. In addition, it makes the user (or a requesting component) unaware of the original location of data (i.e. a remote repository). This is accomplished by an interface abstraction with querying capabilities.

This approach is illustrated in the following figure:



The two types of data sources (local and remote) share the same interface, the location of the data being transparent for the TAO API. It is the implementation of the data source that takes care of properly connecting to the repository and querying and retrieving the data products. The platform can thus seamlessly access data from local repositories, such as the Sentinel-1 and Sentinel-2 repositories found on the DIAS platforms (CreoDIAS, Mundi and Onda), but also from remote repositories, such as the USGS Landsat repository or Alaska Satellite Facility.

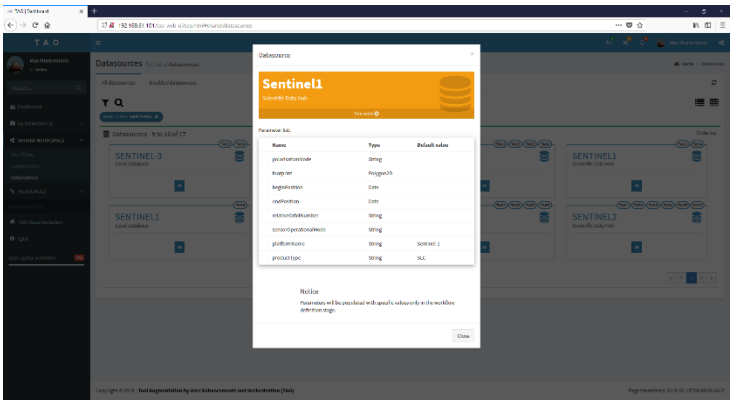
The data source interface exposes operations (i.e. methods) allowing to:

- Connect to the remote repository;
- Authenticate to the remote repository;
- Create a query to be executed against the repository.

Given the diversity of EO products (and product providers), the parameters of a query may be bound to a specific repository. Nevertheless, there is a small subset of parameters that are supported by different providers, namely:

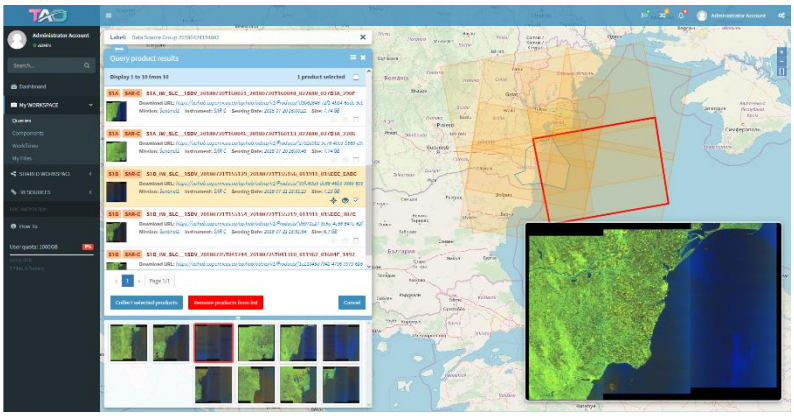
- The product name or identifier;
- The acquisition date (and time);
- The product footprint.

Besides these parameters, various providers may exhibit different parameters (even if a parameter would conceptually represent the same measure, two providers may have different notations for it). In order to cope with this diversity, the query allows for name-value collections of parameters that are described for each data source plugin. This permits adding (or describing) new data sources without any impact (or change) of the object and relational model, using a plugin-based mechanism.



Currently, several data source plugins are available in the TAO platform, namely:

Plugin for	Supported Sensors	Remarks
Scientific Data Hub	Sentinel-1, Sentinel-2, Sentinel-3	Supports also local archives
Amazon Web Services	Sentinel-2, Landsat-8	Supports also local archives
PEPS	Sentinel-1, Sentinel-2	
USGS	Landsat-7, Landsat-8	Supports also local archives
FedEO	ALOS, ALOS-2, CryoSAT, MetOP, DEIMOS-1, ERS-1, ERS-2, ENVISAT, FORMOSAT-2, GeoEYE-1, GOES, IKONOS, IRS-1D, IRS-P5, IRS-P6, JASON, KANOPUS-V1, KOMPSAT-2, METEOR-3M, OCEANSAT, OCEANSAT-2, Pleiades, Proba-V, QuickBird, RadarSAT-1, RapidEYE, SPOT 1-5, SeaSAT, TerraSAR-X, WorldView-1, WorldView-2,	Collections limited to ESA archive. Not all products may be retrieved
Alaska Satellite Facility	Sentinel-1, ALOS	
CreoDIAS	Sentinel-1, Sentinel-2, Landsat-8	Supports also local archives
Mundi DIAS	Sentinel-1, Sentinel-2, Landsat-8	Supports also local archives



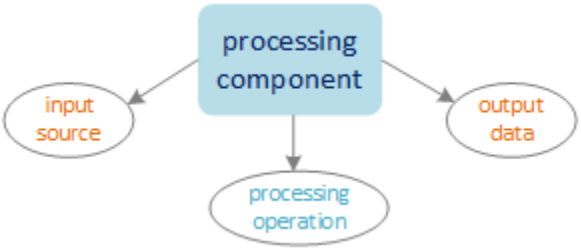
All data sources can be intuitively queried by means of the TAO web interface, which dynamically adapts to the parameters of the respective data source. The query results can be then selectively downloaded, and used as input for further processing.

PROCESSING COMPONENTS AND MODULES

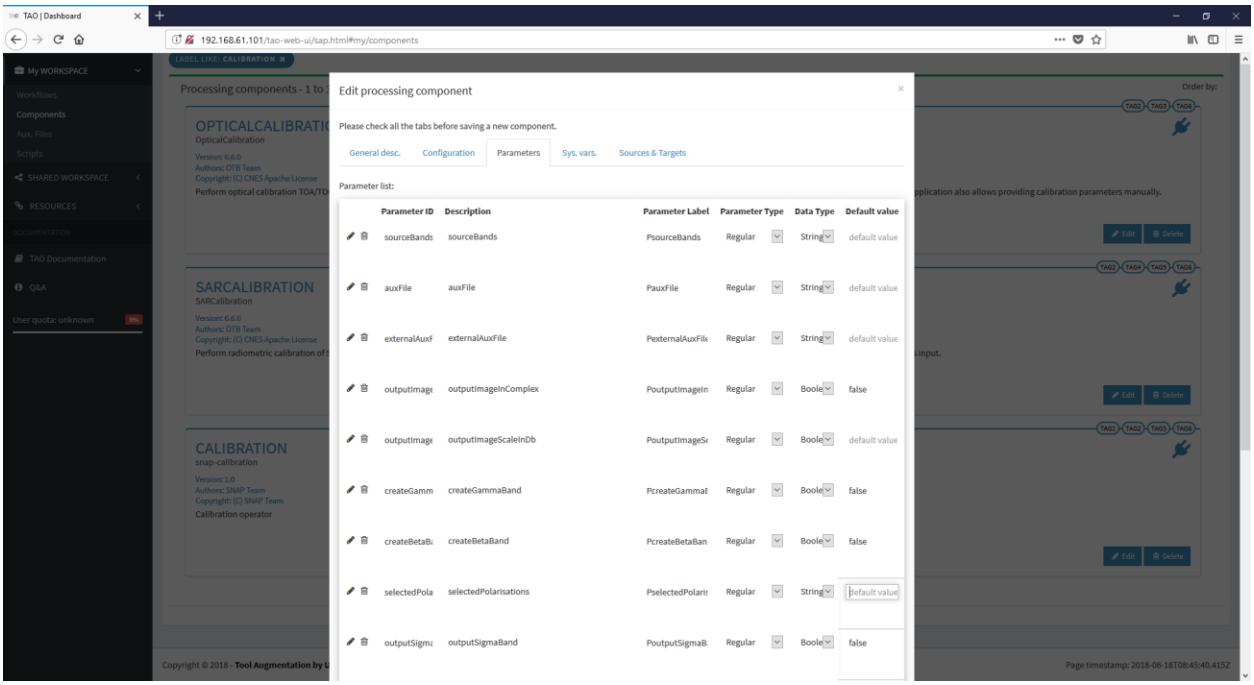
A **processing component** (on short called component) represents a standalone **application** (or module) defined by the following parameters:

- Input description: type of data that the component accept as input source (e.g. image, raster maps, vector maps, sensors, etc.)
- Processing operation with execution parameters: the operation that the component executes with the list of accepted parameters.
- Output description: type of data provided as processing operation result. Can consist in one or more files.

A component is viewed as a generic execution resource and should permit, through its parameters list, the possibility to define *input location* (the place where it expects to retrieve input data for consuming) and *output location* (the place where the processing results are persisted).



Each processing component is described in terms of its expected input, processing module (see the below paragraphs for details about what a processing module is) and output.



There are three kinds of components that could be used in a workflow:

- **System components**: these are the components that ship with the system and can be used by everyone. Currently, TAO ships with all the components of Orfeo Toolbox 6.4 and of SNAP 6.0.0.
- **User components**: a user of the platform also has the possibility to upload his own processing algorithms as script in Python or R. These user components can be used only by the owner (private mode) or by everyone (public mode - in which case it will be part of a contributor list). Users who upload components into the system are responsible for the management of their components.
- **Contributor components list**: all user components loaded into the system and declared for public use will be a part of this list.

The person who upload a component into the system is responsible for providing also all the necessary dependencies. The component is saved into a repository either as an archive or as an installation kit. The repository can be a tree folder into the file system or a database.

The processing modules are heterogeneous regarding their implementation language or operating system. The TAO framework does not target to integrate at once the processing modules from several operating systems (i.e. multiple modules deployed on different operating systems at once). Nevertheless, TAO aims to be as OS-independent as possible and to allow the integration of modules written in different programming languages (such as C/C++, Java or Python).

In an ideal scenario, a processing module may be an independent executable (i.e. not having any additional runtime dependencies). However, this is seldom the case and different modules have different runtime dependencies. Therefore, it is critical that each module has its own appropriate dependencies when the module would execute. Moreover, dependencies of one module may break the execution of another module if they are executed on the same machine, in the same memory space.

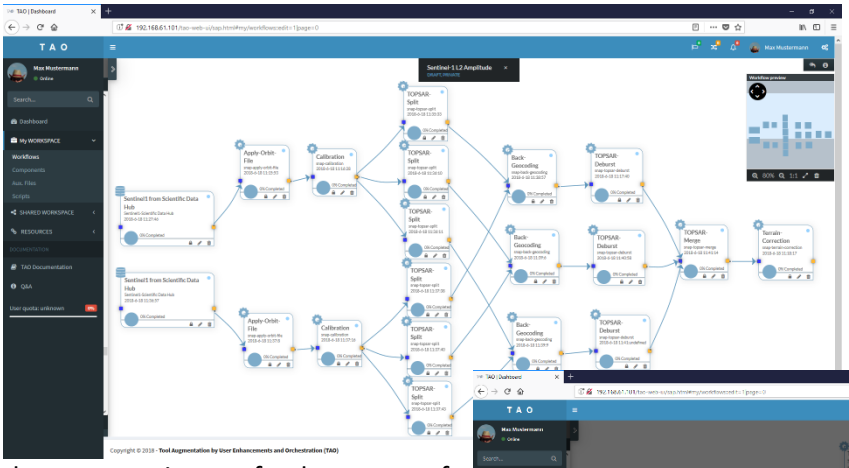
With these constraints in mind, the framework is capable of providing to each module its appropriate runtime dependencies, without disrupting the functioning of other modules, by using Docker execution containers. Each container hosts a processing module together with its runtime dependencies. Each container is then isolated from other containers that may execute on the same machine.

Out of the box, TAO ships with predefined Docker images for OTB 6.4.0, SNAP 6.0.0, Python (with most common python scientific libraries) and R. Of course, additional Docker images can be registered if available.

WORKFLOWS

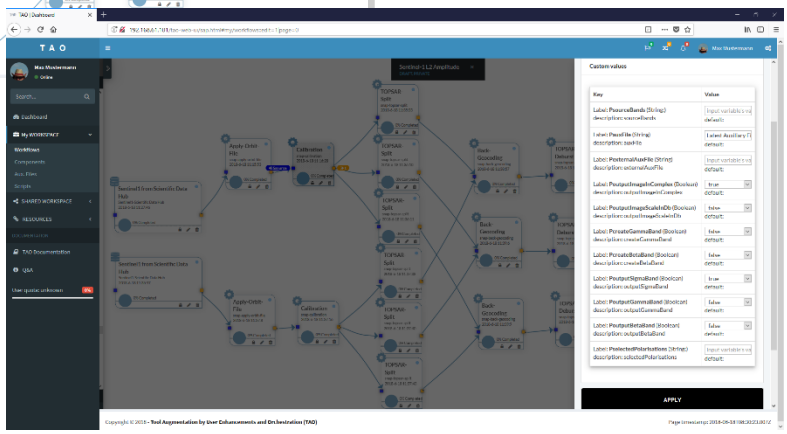
By **workflow** it is understood a sequence of processing operations performed on a given input (EO data and/or ancillary data), having at least one output. A workflow is described by a formal flow diagramming technique, with directed flows between components.

The processing operations in a workflow are handled by processing components.



The Workflow Management feature of TAO copes with all operations necessary to define such workflows and parameters of their components.

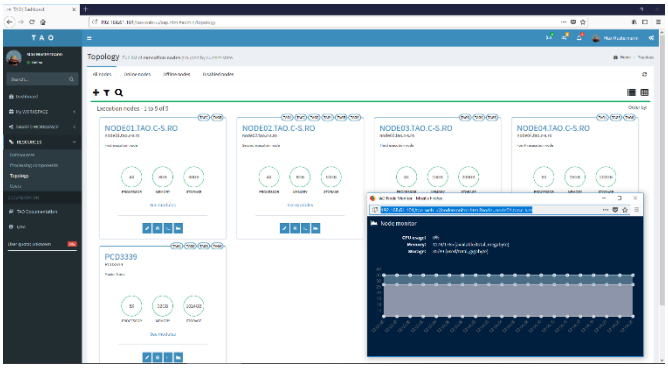
After defining / describing processing components, these can be glued together to form a workflow. A workflow definition then comprises of the set of constituent modules and the rules that link them.



For a better user experience, a workflow is graphically / visually created by dragging and dropping processing components (boxes) on a drawing canvas, and then connecting them with directed (arrowed) lines. Then, for each component, the parameters of the component can be modified.

A workflow can be *created* from scratch, can be *edited* or *deleted*, and/or can be created (*cloned*) from an existing workflow.

COMPUTING RESOURCES AND EXECUTION

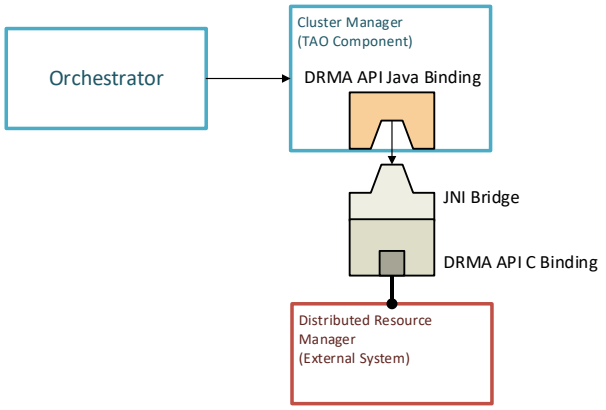


The TAO platform can scale up from one to as many nodes as made available. A node represents a computer resource (a machine) with a defined amount of processing power (number of processors, amount of RAM and disk space), a container for execution of processing components. The platform provides a dedicated administrative interface to manage the processing nodes (even via remote shells).

Currently, the framework deals with already existing nodes (Linux-based with default installation – i.e. no particular configuration) that can be registered with it. There is work in progress of integrating the OpenStack Nova API (used by several cloud providers) to allow the dynamic creation and registration of computing machines/resources). Once done, this will allow the dynamic expansion of resources for achieving computation scalability as needed, without any operator intervention.

The actual execution of workflows is handled by dedicated software, usually named Cluster Resource Managers. In order to decouple the core of the TAO framework from the actual Cluster Resource Manager software, the latter component has to provide a DRMAA-compliant interface.

The DRMAA (Distributed Resource Management Application API) provides a standardized access to the DRM systems for execution resources. It is focused on job submission, job control, reservation management, and retrieval of jobs and machine monitoring information.



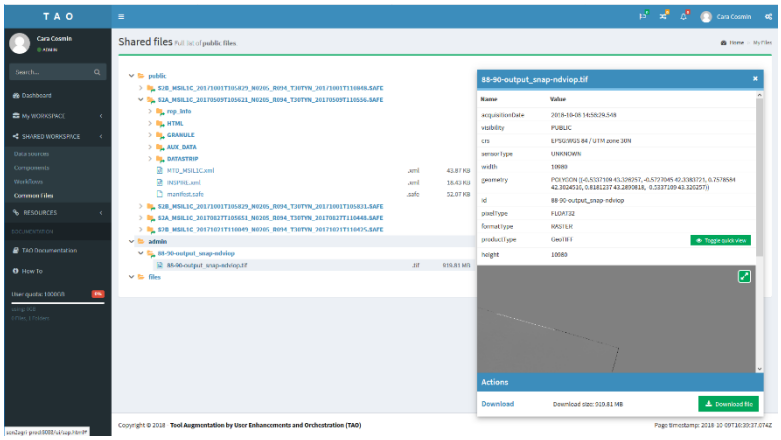
Currently, there are several DRMAA implementations, most of them implemented in the C programming language and for Linux operating systems.

The bridging plugin approach taken by the TAO framework allows to easily swap such implementations, making the framework loosely coupled from the actual DRM system used.

The platform provides two such DRMAA-compliant plugins, namely for SLURM (Simple LinUx Resource Manager) and Torque (the open-source version of PBS/Torque).

USER WORKSPACES

The results of the workflow executions (and not only) can be found in the user workspace. This is a system view that allows a user to:

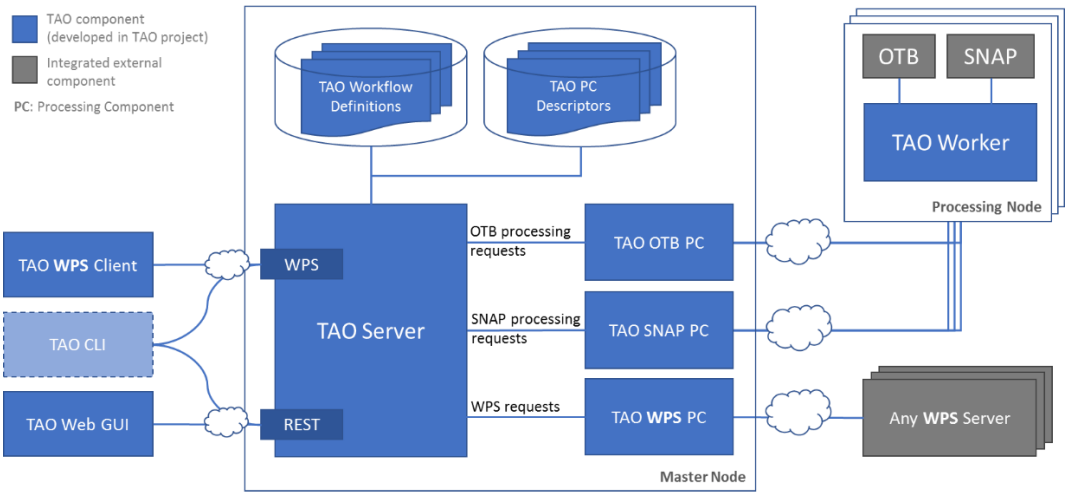


- View the details about a local data product (accessible for both administrator and user roles, with the difference that the user can only view details about his local data products as well as public data products),
- View all local data products (accessible for both administrator and user roles, with the difference that the user can only view his local data products as well as public data products),
- Upload additional files that can be further used in workflows (such as model files, shape files, etc.),
- Perform a check-style selection from the existing products that can be used as input for his/her workflows

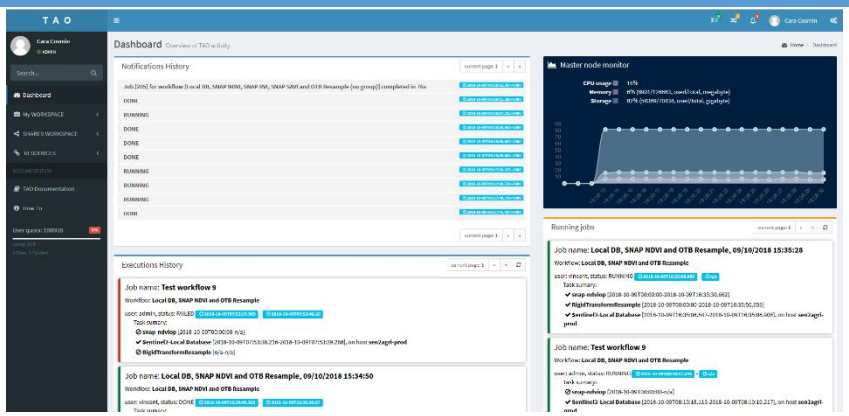
EXTERNAL PLATFORM ACCESS

A closed platform is of limited use and impales its adoption by the community. Therefore, the TAO platform allows for the interoperation with external processing systems, and also for a standardized access by external clients. To accomplish this, it relies on several OGC WPS – compliant components, each having a specific role:

- **WPS processing component**: this is the TAO component that can invoke external processors via a WPS interface;
- **WPS server**: this is the TAO component that allows external clients to invoke TAO workflows. The other TAO processing components are not exposed via WPS interfaces. Instead, only public workflows are exposed as WPS capabilities of this WPS interface.



MONITORING



The platform comes with a minimal, but sufficient monitoring dashboard, which allows, at any time, to see what is executing in the system and where it is executing the system resource usage (CPU, memory, storage space).

Besides platform information, users and administrators are visually notified, in real-time, via the web interface, about events occurring in the system.

Currently, the monitoring dashboard presents the following indicators:

Name	Purpose	Type
Status of an execution node	Indicates the status of a particular execution node	Possible statuses: online, offline, not configured.
Workflow status	Indicates the status of the execution of a particular workflow	Possible statuses: not run, running, paused, completed, error.
Task status	Indicates the status of the execution of a particular workflow task	Possible statuses: not run, running, completed, error.
Quota status	Indicates the status of the user quota	Percentage and value of used disk space.