ESCAPE

European Science Cluster of Astronomy & Particle physics ESFRI research Infrastructures

WP3 – ESCAPE Open-source Software and Services repository

Technical overview, 28-10-2021,

Thomas Vuillaume, Enrique Garcia*

ESCAPE - The European Science Cluster of Astronomy & Particle Physics ESFRI Research Infrastructures has received funding from the European Union's Horizon 2020 research and innovation programme under the Grant Agreement n° 824064.

Outline

- OSSR goals
- Entry point / OSSR portal
- OSSR itself
  - Onboarging process and status.
  - EOSSR library.
  - OSSR Metadata.
  - CI and implementation intro OSSR.

Funded by the European Union's Horizon 2020 - Grant N° 824064

Goals of the OSSR

1. Provide software and services following the **FAIR*** principles:
   - **Findable** : Rich metadata, **unique** and **persistent** identifier
   - **Accessible** : Metadata and data are **understandable** to **humans** and **machines**. Data is deposited in a **trusted repository**.
   - **Interoperable** : Metadata use a formal, accessible, shared, and broadly applicable language for knowledge representation.
   - **Reusable** : Data and collections have a **clear** usage **licenses** and provide accurate information on **provenance**.

2. Provide the infrastructure and services to foster FAIR (effortless) contributions
   - **ESCAPE Virtual Environment** (see Arturo's presentation)

Funded by the European Union's Horizon 2020 - Grant N° 824064

OSSR entry point

OSSR portal

http://purl.org/escape/ossr

You should start here

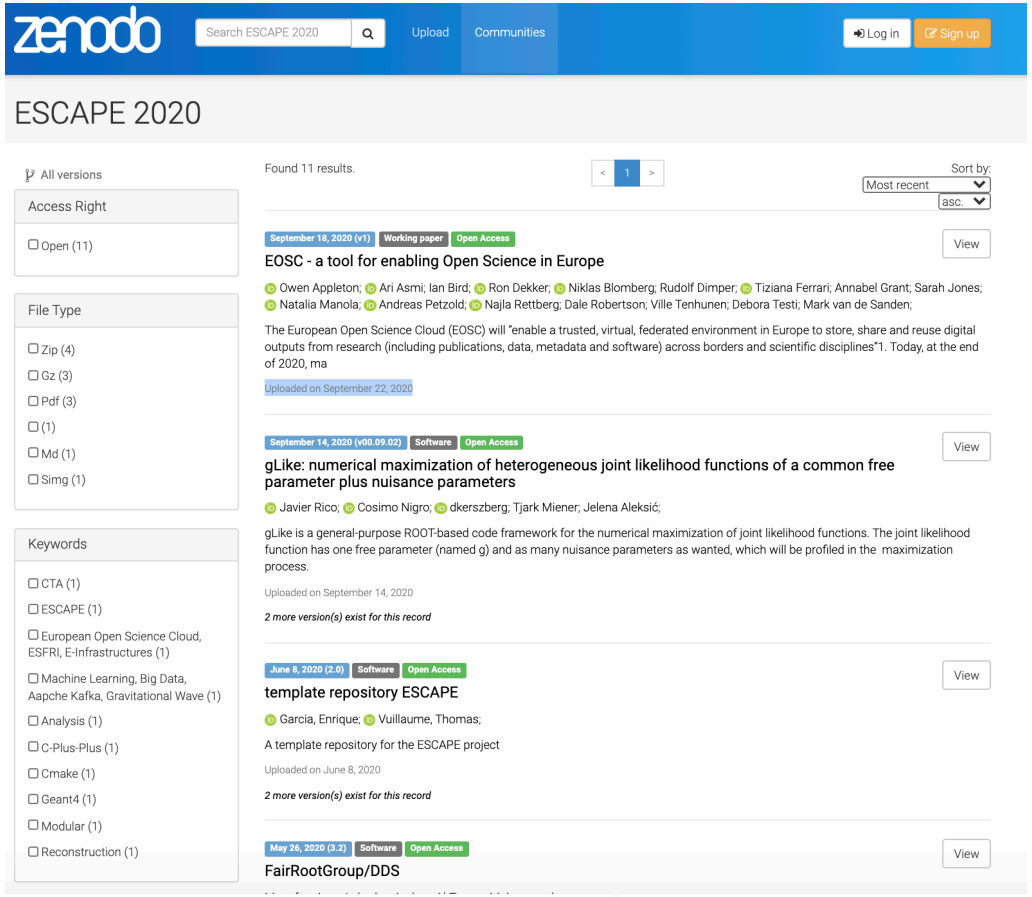
- Browse and search by projects
- Find all the required information to use the services and contribute to the OSSR
- Tutorials, tools...
- Accessible from https://projectescape.eu/

Funded by the European Union's Horizon 2020 - Grant N° 824064

OSSR technology choice: Zenodo community

https://zenodo.org/communities/escape2020/

- General purpose repository developed by the OpenAIRE program.
- Maintained and hosted by CERN.
- Follows and enforces FAIR principles.
- Trusted repository.
- Provides web interface with search and filters.
- Provides api for automated requests.



Funded by the European Union's Horizon 2020 - Grant N° 824064

Curation and onboarding process

- Guidelines
  - Part of deliverable D3.7 *License, provenance and metadata guidelines for the software and service repository*
  - The digitial resource must contain a license
    - Preferably a persmissive and open-source one
  - Minimal metadata must be provided
    - Preferably through a codemeta.json file (more on that later)

- Checklist and procedure
  - Code source and stable release
  - License
  - Virtualisation container / image (optional)
  - Codemeta.json file
  - Registration form
    - Registered onboarding presentation during WP3 FG1 call
    - Technical report
  - Upload to OSSR: Zenodo escape2020 community
    - Curation process and acceptance if meets all the requirements and procedure is complete

Funded by the European Union's Horizon 2020 - Grant N° 824064

Onboarding status

Accessible from the OSSR portal

- 15 on-going onboarding
- Most of which are at 80% and need a little push / reminder to get done
- Dedicated pages produced automatically per project on the portal

#	Tracker	Statut	Priorité	Sujet	Mis-à-jour
122	Integration	In Progress	Normal	Onboarding: ATLAS Open Data C++ analysis software at 13 TeV	17/09/2021 09:10
121	Integration	New	Normal	Onboarding: UNITOV services	09/09/2021 09:24
120	Integration	New	Normal	Onboarding: LOFAR software	08/09/2021 21:32
42	Integration	New	Normal	Onboarding: R3B	26/05/2021 17:53
37	Integration	In Progress	Normal	Onboarding: gLike	09/07/2021 08:42
36	Integration	In Progress	Normal	Onboarding: km3py	09/07/2021 08:42
35	Integration	In Progress	Normal	Onboarding: gammapy	09/07/2021 08:43
34	Integration	In Progress	Normal	Onboarding: HCG-16 study	09/07/2021 08:43
33	Integration	In Progress	Normal	Onboarding: agnpy	09/07/2021 08:48
32	Integration	In Progress	Normal	Onboarding: ConCordia	09/07/2021 08:49
31	Integration	In Progress	Normal	Onboarding: FAIR software	09/07/2021 08:49
30	Integration	In Progress	Normal	Onboarding: SKA data challenge	09/07/2021 08:49
27	Integration	In Progress	Normal	Onboarding: JIVE software	09/07/2021 08:50
25	Integration	In Progress	Normal	Onboarding: Gammalearn	09/07/2021 08:51
8	Integration	New	Normal	[TEMPLATE] Onboarding: software or container	14/09/2021 10:54

Funded by the European Union's Horizon 2020 - Grant N° 824064

OSSR entry categorization (draft)

- Scientific analysis software
  - fully installable software packages, e.g. gammypy
- Service environments
  - Projects for data and service providers, which offer e.g. a platform, GUI, or middleware for the computing environment (e.g. ConCordia)
- dedicated scientific products (analysis, data)
  - **repositories of data or workflows**: projects that provide a certain class of data objects and serve as repository for this class of data, e.g. jupyter notebooks, or certain analysis results (if not provided in datalake or VO), e.g. CME database
  - **self-contained analysis environments**: repositories which provide full workflows and data (access), e.g. to reproduce a given analysis (e.g. SKA data challenge, HCG-16 study)

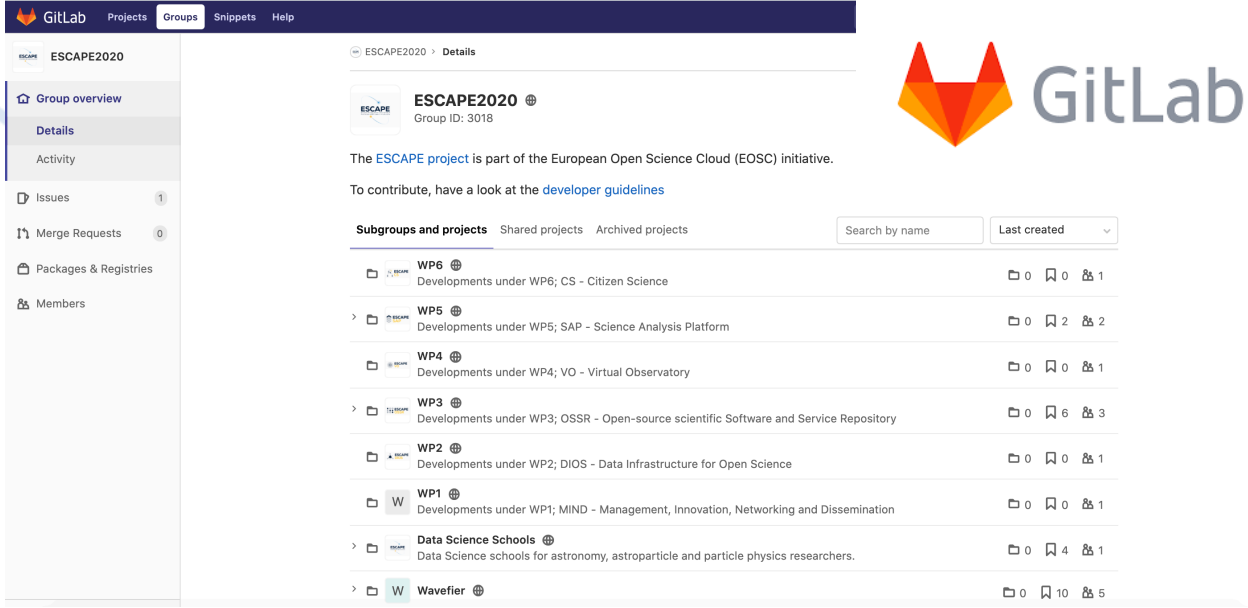
Can be loaded in ESAP?

ESCAPE services to OSSR?

integration into ESAP?

Funded by the European Union's Horizon 2020 - Grant N° 824064

Tools



- Development platform : https://gitlab.in2p3.fr/escape2020/

- EOSSR library
  - Dev: https://gitlab.in2p3.fr/escape2020/wp3/eossr
  - Doc: https://escape2020.pages.in2p3.fr/wp3/eossr/
  - Regroup all previous and current OSSR developments
  - Python3
  - OSSR API : send request to the OSSR, find and filter software and services, upload new entries, update existing entries
  - CI : automated upload / update using gitlab CI
  - Metadata : schema definition, crosswalk between CodeMeta and Zenodo
  - Should be used by other (automatised) services to access the OSSR

```
pip install eossr==0.2
```


Funded by the European Union's Horizon 2020 - Grant N° 824064

eossr walkthrough

- `eossr.api.Record`
  - https://escape2020.pages.in2p3.fr/wp3/eossr/examples/ossr_api-Explore_the_OSSR.html

> DEMO <

- `eossr.api.zenodo.ZenodoAPI`
  - Manage user entries
    - upload new entry
    - modify/update existing ones
    - used by the CI to upload entry based on CodeMeta.json

Funded by the European Union's Horizon 2020 - Grant N° 824064

OSSR Metadata

- CodeMeta Project
  - implemented by adding codemeta.json file at the root of the project.
- Defines a standard metadata syntax (schema),
- designed to describe important information about software (license, purpose, authors, dependencies…) to facilitate discovery, adoption, and credit.
- Based on Schema.org developed by major search engines (Google, Bing, Yahoo)
- Can be easily crosswalked to other metadata schemas if needed.
- Used by other services such as Software Heritage
- Can be expanded/refined if needed (starting with Schema.org `Type`s)
- Ongoing discussions about extending schema
  - Container metadata
  - Indicate which service can run the entry

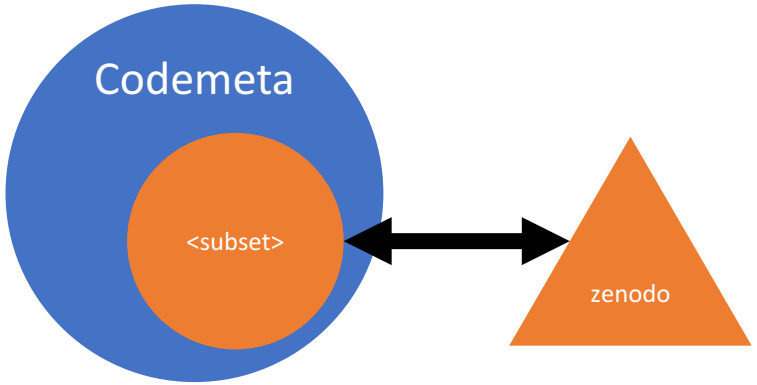
Funded by the European Union's Horizon 2020 - Grant N° 824064

Generate your CodeMeta file

- codemeta.json file at the root of the project
- online generator: https://codemeta.github.io/codemeta-generator/
  - a similar custom escape codemeta generator could be built if we extend the schema. At the moment, use the official one.
- Generate it using existing tools from your already existing package (in Ruby, Python, R)
  - WIP: from GitHub repository
- by hand, following the schema describe here or here (not recommended, unless you want to add metadata not covered by these tools)

Funded by the European Union's Horizon 2020 - Grant N° 824064

Note on Zenodo metadata



- Zenodo does **not** use codemeta
  - it uses an internal specific metadata schema, implemented in a .zenodo.json file at the root of the entry
  - you can add the .zenodo.json file yourself to provide all information to Zenodo, or use Zenodo's web interface to provide the necessary metadata when creating a record
  - the CI uses eossr codemeta2zenodo (developed in WP3) to crosswalk codemeta.json into .zenodo.json
  - Zenodo is exploring the possibility to use / read codemeta directly

- When interrogating the OSSR through Zenodo API, we are limited to zenodo metadata **at first**
  - but once an entry has been found, we can retrieve the codemeta.json alone and thus have access to the more complete metadata
  - With eossr library: `` `Record.get_codemeta()` ``

Funded by the European Union's Horizon 2020 - Grant N° 824064

Going beyond CodeMeta

- The issue has been raised (here, here) that we (in particular ESAP) might have specific queries needs (notebooks, containers) to identify OSSR entries that are not (at least not clearly) implemented in CodeMeta or Schema.org schemas

- The issue is not new and we are not alone (see discussions from 2018 to extend schema.org to containers). Spoiler alert: the discussion is still open…

- But we don't need to solve the problem for the entire world right now, let's solve it for us ☺. Possibilities;
  - extend schema
  - use specific keywords

- Opened points and discussion in the eossr issues

Funded by the European Union's Horizon 2020 - Grant N° 824064

Continuous Integration at your service

- Uses codemeta.json in the code repository and eossr library
- at the moment using GitLab CI (WIP: using github actions)
- Upon software release:
  - update OSSR entry (through api)
  - build a container
    - Singularity or Docker image → can be added into Zenodo entry
    - Docker containers → added to the gitlab registry (acts as docker hub)

- See the ESCAPE template project for a working example

Funded by the European Union's Horizon 2020 - Grant N° 824064

Implementation into the OSSR environment

AUTOMATISED

- Publishes source code (updates your existing record with new versions)

zenodo

- Long term archived
- Findable
- Citable

GitLab

1. Make a new tag (release)
2. Let the CI do the rest

- builds images

- publishes to OSSR

- Publishes on registries

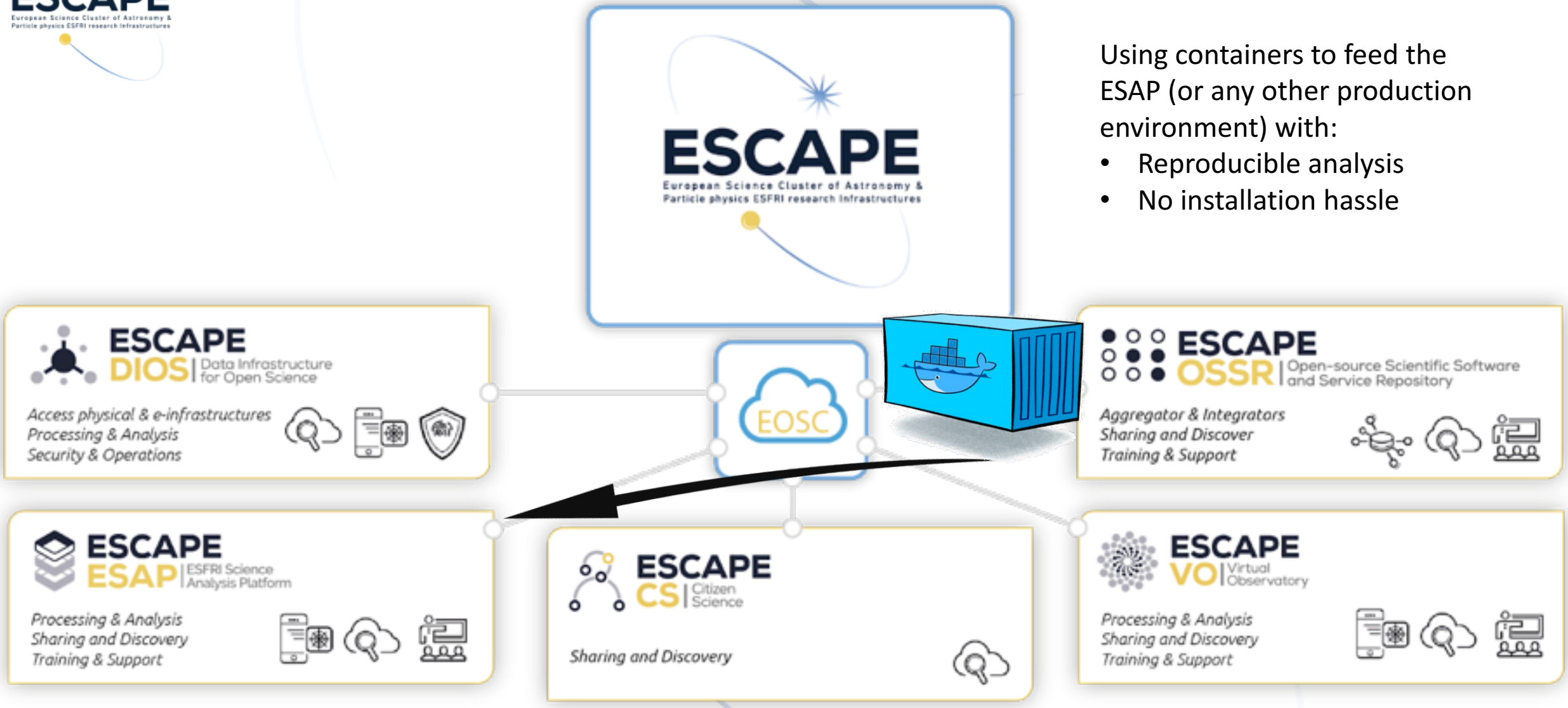
GitLab registry

docker hub

Funded by the European Union's Horizon 2020 - Grant N° 824064

Using containers to feed the ESAP (or any other production environment) with:

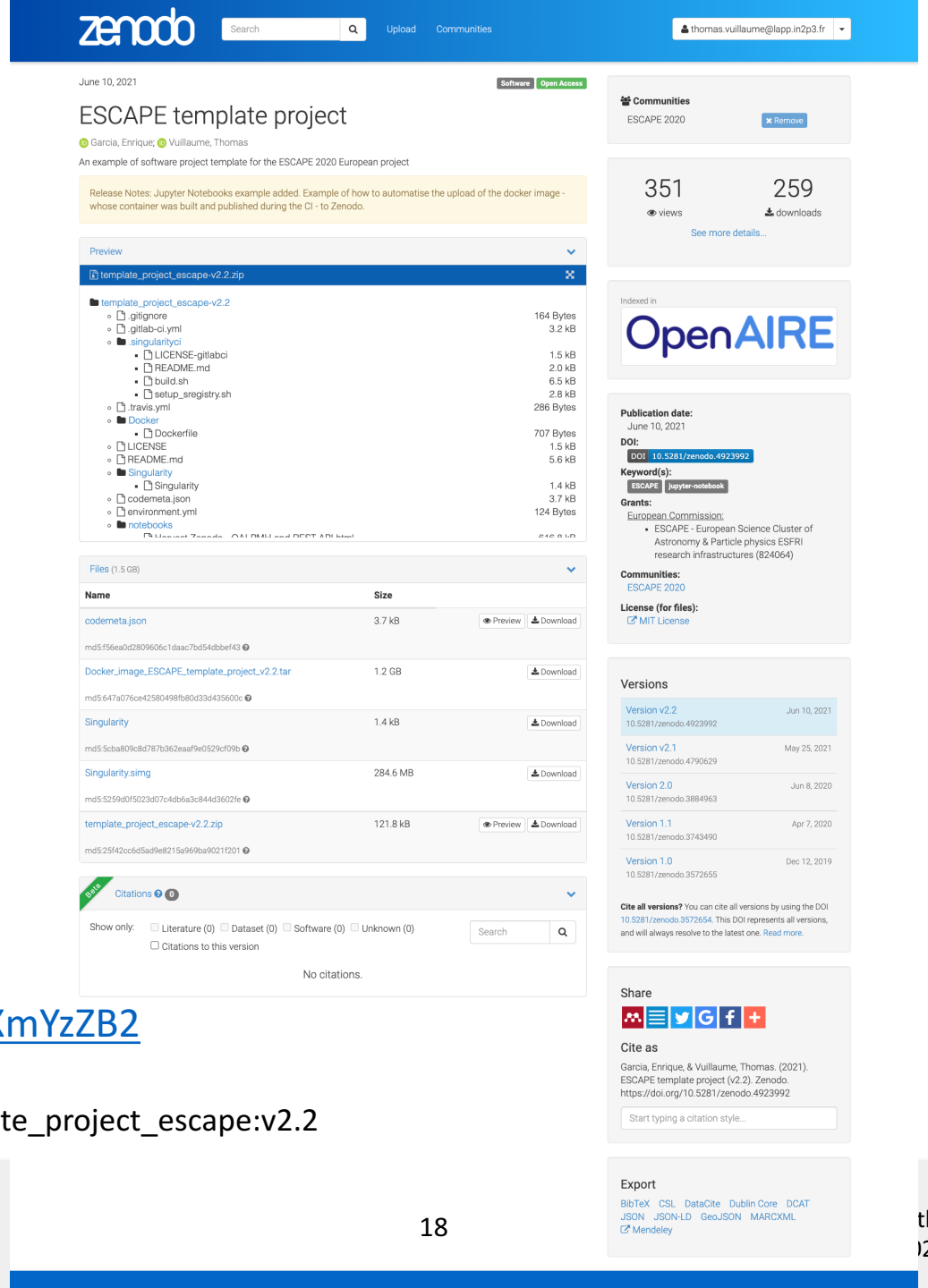
- Reproducible analysis
- No installation hassle



Funded by the European Union's Horizon 2020 - Grant N° 824064

OSSR record example:
ESCAPE template project

- Source code preview
- Record content
- Cited by



- Communities
- Views and downloads
- DOI: persistent identifier
- keywords
- Funding
- License
- Versions
- Share

and cite

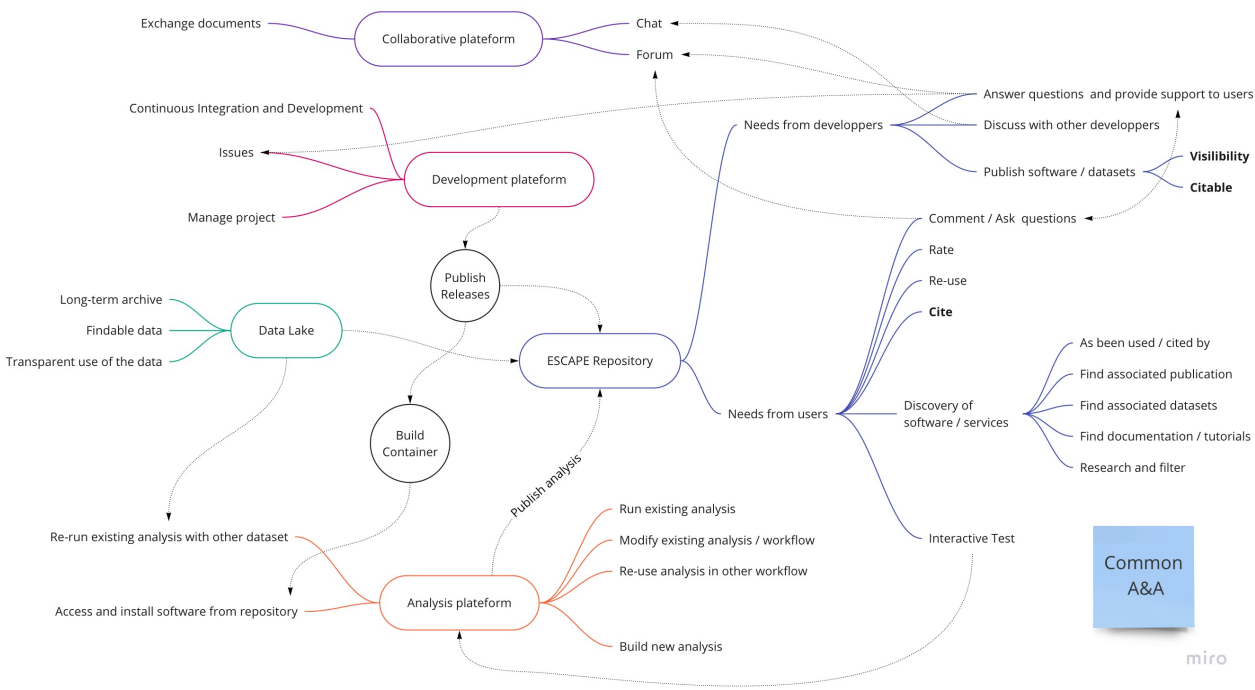
https://zenodo.org/record/4923992#.YUxKXmYzZB2

Docker: gitlab-registry.in2p3.fr/escape2020/wp3/template_project_escape:v2.2

the European Union's
020 - Grant N° 824064

Conclusion

Aim: ESCAPE virtual environment



- A lot of technical developments to create an integrated environment and ease the use of the OSSR
- WIP: Connection with other services
  - see WP3-WP5 discussion
- WIP: Integrating the OSSR in the research virtual environment
  - See Arturo's presentation
- Discussions and contact point:
  - https://escape2020.pages.in2p3.fr/wp3/ossr-pages/page/contact/

Funded by the European Union's Horizon 2020 - Grant N° 824064

Appendix

Funded by the European Union's Horizon 2020 - Grant N° 824064

OSSR entry categorization (draft)

- Scientific analysis software
  - fully installable software packages, e.g. gammypy
- Service environments
  - Projects for data and service providers, which offer e.g. a platform, GUI, or middleware for the computing environment (e.g. ConCordia)
- dedicated scientific products (analysis, data)
  - **repositories of data or workflows**: projects that provide a certain class of data objects and serve as repository for this class of data, e.g. jupyter notebooks, or certain analysis results (if not provided in datalake or VO), e.g. CME database
  - **self-contained analysis environments**: repositories which provide full workflows and data (access), e.g. to reproduce a given analysis (e.g. SKA data challenge, HCG-16 study)

Can be loaded in ESAP?

ESCAPE services to OSSR?

integration into ESAP?

Funded by the European Union's Horizon 2020 - Grant N° 824064

An example of target project : The Crab bundle

The Crab multi-instrument gamma-ray analysis with MAGIC, VERITAS, FACT and H.E.S.S.
https://zenodo.org/record/2381863#.XkxcD5NKhhA
https://github.com/open-gamma-ray-astro/joint-crab/tree/v0.1

December 18, 2018 Software Open Access

The joint-crab bundle

C. Nigro; C. Deil; R. Zanin; T. Hassan; J. King; J.E. Ruiz; L. Saha; R. Terrier; K. Bruegge; M. Noethe; R. Bird; T. T. Y. Lin; J. Aleksic; C. Boisson; J.L. Contreras; A. Donath; L. Jouvin; N. Kelley-Hoskins; B. Khelifi; K. Kosack; J. Rico; A. Sinha

This `joint-crab` bundle allows for a first reproducible multi-instrument gamma-ray analysis, achieved by using the prototypical DL3 data format and the open-source Gammapy software package, for a small set of MAGIC, VERITAS, FACT, and H.E.S.S. Crab nebula observations.

Preview

joint-crab-v0.1.zip

open-gamma-ray-astro-joint-crab-752a165	
.gitignore	1.2 kB
1_data.ipynb	3.0 kB
2_results.ipynb	6.0 kB
3_systematics.ipynb	100.8 kB
4_naima.ipynb	72.0 kB
5_crab_pulsar_nebula_sed.ipynb	65.4 kB
Dockerfile	604 Bytes
LICENSE	1.5 kB
README.md	4.3 kB
analysis.md	2.9 kB
binder	
environment.yml	443 Bytes
data	
README.md	512 Bytes
fact	
20131103_103_dl3.fits	25.9 kB
20131103_104_dl3.fits	25.9 kB
20131103_105_dl3.fits	23.0 kB

141 views
12 downloads
See more details...

Indexed in OpenAIRE

Publication date: December 18, 2018
DOI: DOI 10.5281/zenodo.2381863
Keyword(s): Astronomy Gamma-rays Data analysis
Related identifiers: Referenced by https://arxiv.org/abs/1903.06621
Alternate identifiers: https://github.com/open-gamma-ray-astro/joint-crab/tree/v0.1
Communities: Astronomy-General

Files (10.1 MB)

Name	Size	
open-gamma-ray-astro/joint-crab-v0.1.zip	10.1 MB	Preview Download

md5:55cd7fed91c53f54a96e32945e764665

Beta Citations 1

Show only: Literature (1) Dataset (0) Software (0) Unknown (0) Citations to this version

Search

Towards open and reproducible multi-instrument analysis in ... 2019 ADS ARXIV DOI
Nigro, C. et al. (DOI: 10.1051/0004-6361/201834938)

< 1 > Page size: 10

License (for files): BSD 3-Clause Clear License

Versions
Version v0.1 Dec 18, 2018
10.5281/zenodo.2381863

Cite all versions? You can cite all versions by using the DOI 10.5281/zenodo.2381862. This DOI represents all versions, and will always resolve to the latest one. Read more.

Share

Cite as
C. Nigro, C. Deil, R. Zanin, T. Hassan, J. King, J.E. Ruiz, ... A. Sinha. (2018, December 18). The joint-crab bundle (Version v0.1). Zenodo. http://doi.org/10.5281/zenodo.2381863

Start typing a citation style...

Export
BibTeX CSL DataCite Dublin Core DCAT JSON JSON-LD GeoJSON MARCXML Mendeley

license

Cited by

Link to project and article

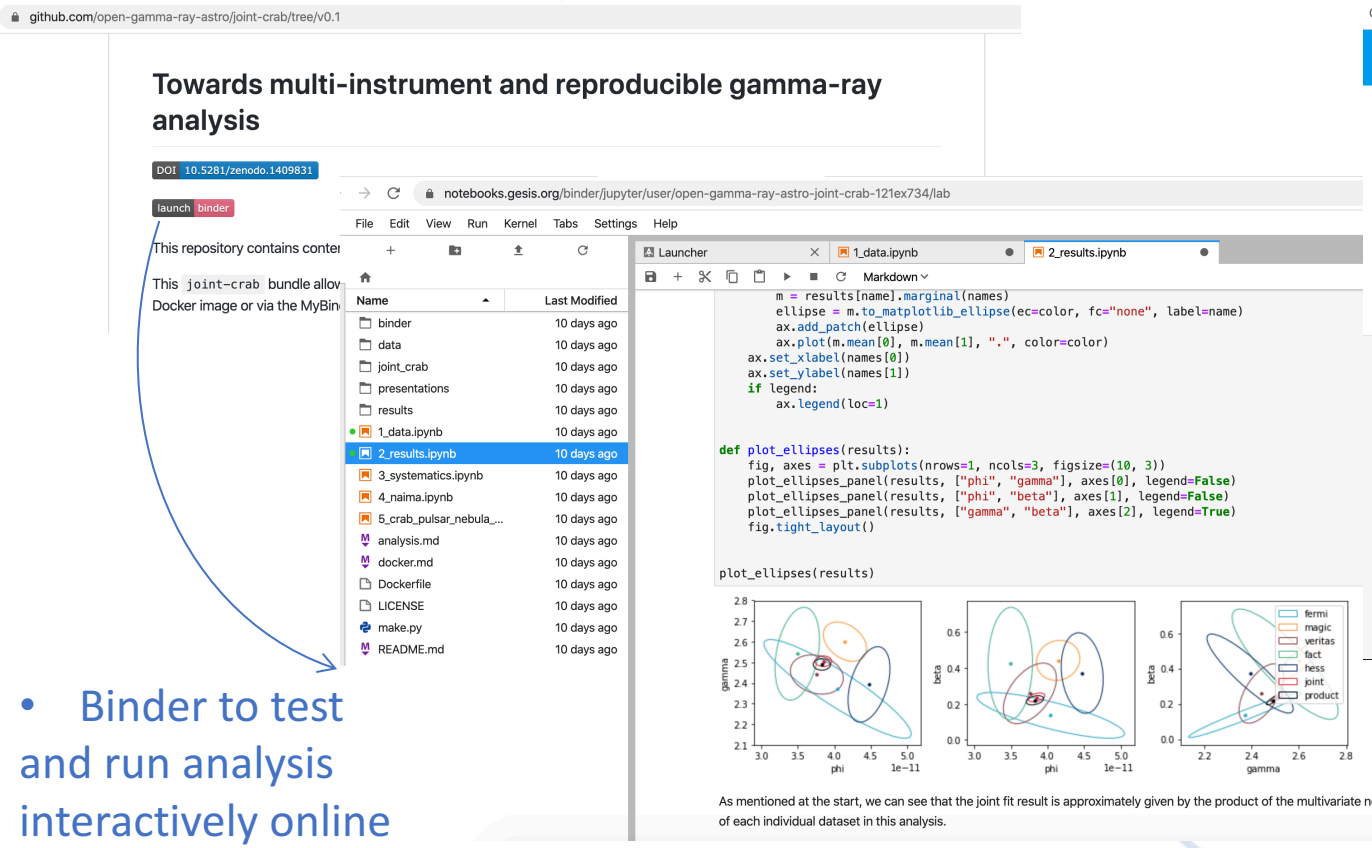
Cite as

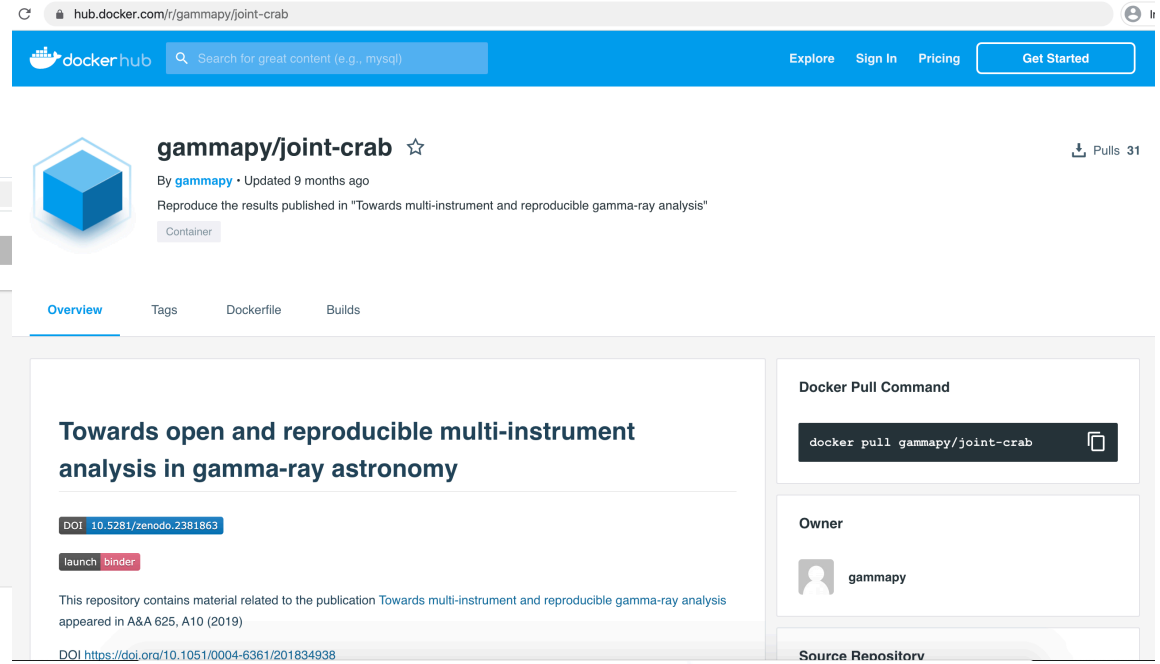
Source code and data

Funded by the European Union's Horizon 2020 - Grant N° 824064

An example of project : The Crab bundle

The Crab multi-instrument gamma-ray analysis with MAGIC, VERITAS, FACT and H.E.S.S.
https://zenodo.org/record/2381863#.XkxcD5NKhhA
https://github.com/open-gamma-ray-astro/joint-crab/tree/v0.1





- Binder to test and run analysis interactively online
- Docker to ensure reproducibility and ease of use

Funded by the European Union's Horizon 2020 - Grant N° 824064