



WP3 – ESCAPE Open-source Software and Services repository

Technical overview, 28-10-2021,
Thomas Vuillaume, Enrique Garcia*



Outline

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



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](#))



OSSR entry point

OSSR portal

<http://purl.org/escape/ossr>











- 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/>



You should start here

Search the ESCAPE repository

Research infrastructures and Science Projects in the OSSR

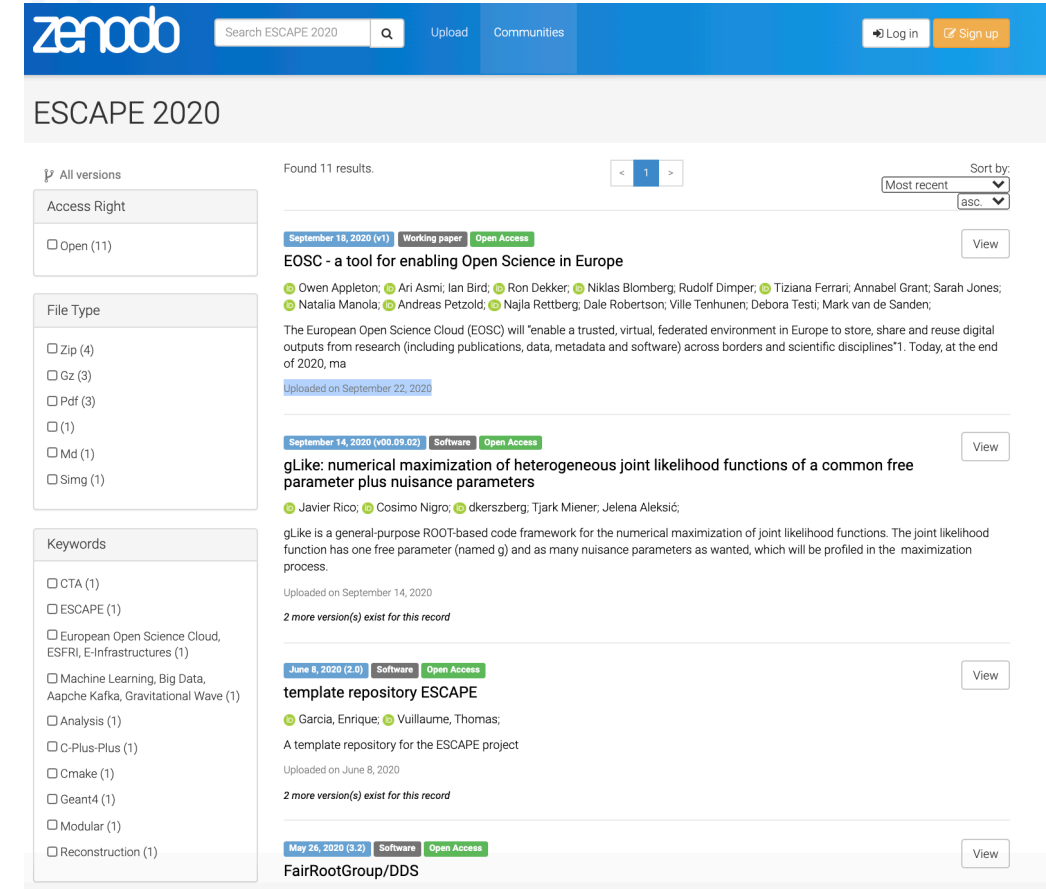
 cta cherenkov telescope array	 LSST Legacy Survey of Space and Time	 VIRGO
 KM3NeT	 ESO	 EST european solar telescope
 SKA SQUARE KILOMETRE ARRAY	 JIVE Joint Institute for VLBI ERIC	 FAIR
 HiLumi		



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.



The screenshot shows the Zenodo website interface for the ESCAPE 2020 community. The header includes the Zenodo logo, a search bar with "Search ESCAPE 2020", and links for "Upload", "Communities", "Log in", and "Sign up". The main content area is titled "ESCAPE 2020" and displays a list of research outputs. On the left, there are filters for "Access Right" (Open (11)), "File Type" (Zip (4), Gz (3), Pdf (3), etc.), and "Keywords" (CTA (1), ESCAPE (1), etc.). The main list shows three items: "EOSC - a tool for enabling Open Science in Europe" (September 16, 2020), "gLike: numerical maximization of heterogeneous joint likelihood functions of a common free parameter plus nuisance parameters" (September 14, 2020), and "template repository ESCAPE" (June 8, 2020). Each item includes a "View" button and a "2 more version(s) exist for this record" link.



Curation and onboarding process

● Guidelines

- Part of deliverable D3.7 *License, provenance and metadata guidelines for the software and service repository*
- The digital resource must contain a license
 - Preferably a permissive 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



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

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



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?

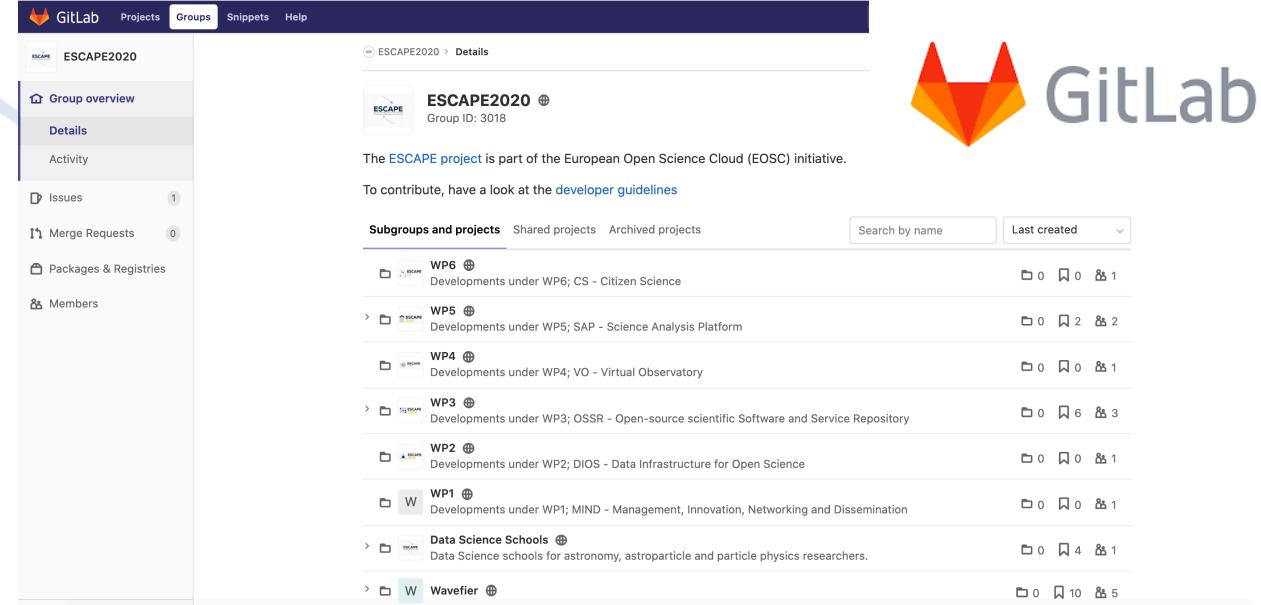
integration
into ESAP?



- 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
```



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



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

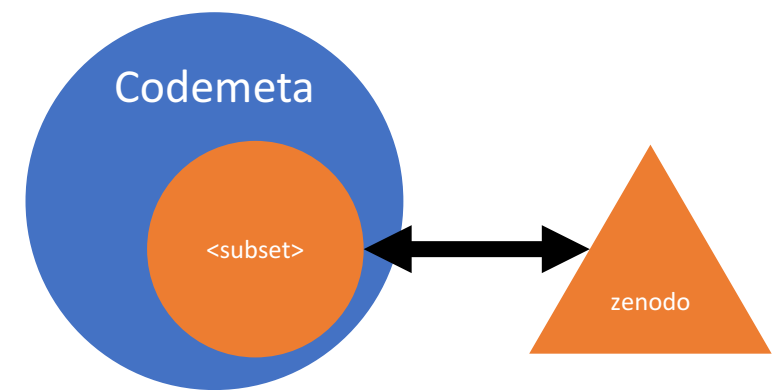


Generate your CodeMeta file

- 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)



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()`



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](#)



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



Implementation into the OSSR environment

AUTOMATISED

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

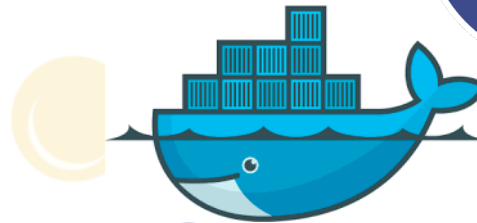


- Long term archived
- Findable
- Citable



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

- builds images



- publishes to OSSR



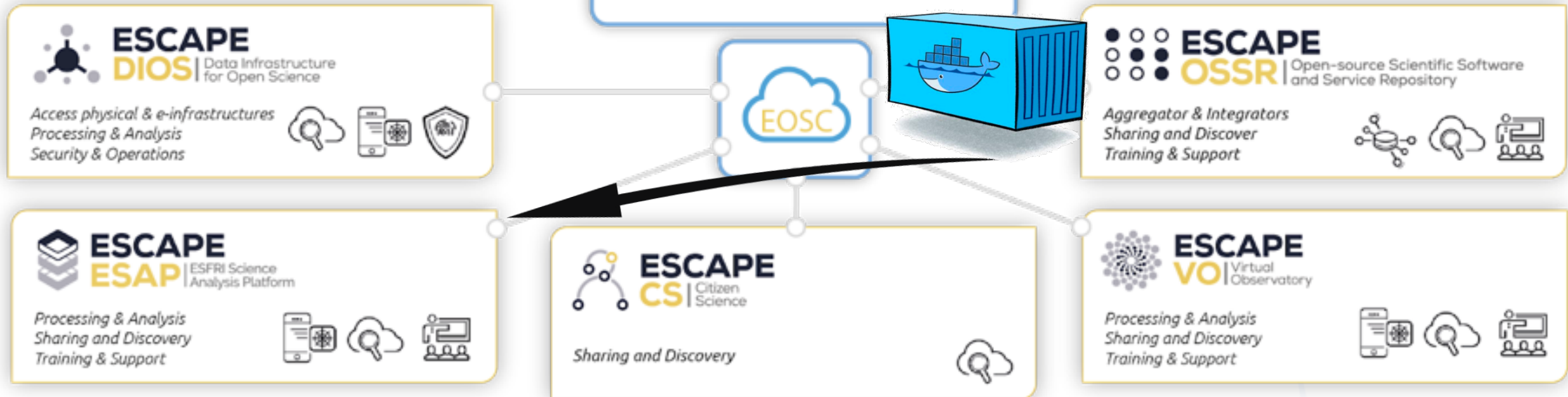
- Publishes on registries





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

- Reproducible analysis
- No installation hassle

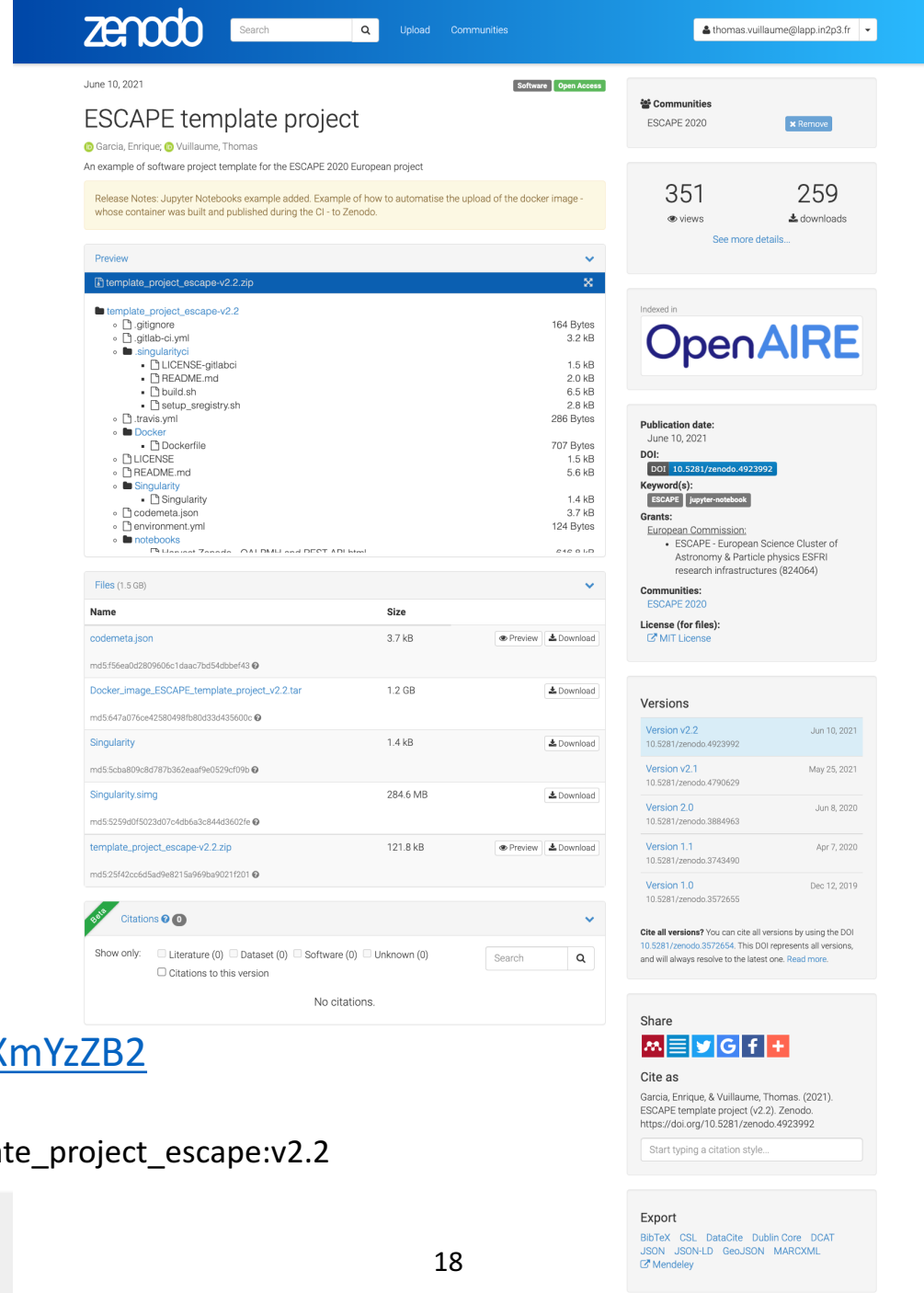


OSSR record example: ESCAPE template project

- Source code preview
- Record content
- Cited by

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

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



June 10, 2021

Software Open Access

ESCAPE template project

Garcia, Enrique; Vuillaume, Thomas

An example of software project template for the ESCAPE 2020 European project

Release Notes: Jupyter Notebooks example added. Example of how to automatise the upload of the docker image - whose container was built and published during the CI - to Zenodo.

Preview

template_project_escape-v2.2.zip

- template_project_escape-v2.2
 - gitignore 164 Bytes
 - gitlab-ci.yml 3.2 kB
 - singularity
 - LICENSE-gitlabci 1.5 kB
 - README.md 2.0 kB
 - build.sh 6.5 kB
 - setup_singularity.sh 2.8 kB
 - travis.yml 286 Bytes
 - Docker
 - Dockerfile 707 Bytes
 - LICENSE 1.5 kB
 - README.md 5.6 kB
 - Singularity
 - Singularity 1.4 kB
 - codemeta.json 3.7 kB
 - environment.yml 124 Bytes
 - notebooks

Files (1.5 GB)

Name	Size	Actions
codemeta.json	3.7 kB	Preview Download
md5f56ea0d280960c1daac7cd54dbbef43		
Docker_image_ESCAPE_template_project_v2.2.tar	1.2 GB	Download
md5647a076ce42580498fb80d33435600c		
Singularity	1.4 kB	Download
md55c8a809c8d787b362eaf9e0529cf09b		
Singularity.simg	284.6 MB	Download
md55259d0f5023d07c4db6a3c844d3602fe		
template_project_escape-v2.2.zip	121.8 kB	Preview Download
md525f42cc6d5ad9e8215a969ba9021f201		

Citations

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

No citations.

Communities

ESCAPE 2020

351 views 259 downloads

See more details...

Indexed in

OpenAIRE

Publication date: June 10, 2021

DOI: [10.5281/zenodo.4923992](https://doi.org/10.5281/zenodo.4923992)

Keyword(s): ESCAPE Jupyter-notebook

Grants: European Commission: ESCAPE - European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures (824064)

Communities: ESCAPE 2020

License (for files): MIT License

Versions

Version	Date
Version v2.2	Jun 10, 2021
Version v2.1	May 25, 2021
Version 2.0	Jun 8, 2020
Version 1.1	Apr 7, 2020
Version 1.0	Dec 12, 2019

Cite all versions? You can cite all versions by using the DOI [10.5281/zenodo.3572654](https://doi.org/10.5281/zenodo.3572654). This DOI represents all versions, and will always resolve to the latest one. [Read more.](#)

Share

Cite as

Garcia, Enrique, & Vuillaume, Thomas. (2021). ESCAPE template project (v2.2). Zenodo. <https://doi.org/10.5281/zenodo.4923992>

Start typing a citation style...

Export

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

Mendeley

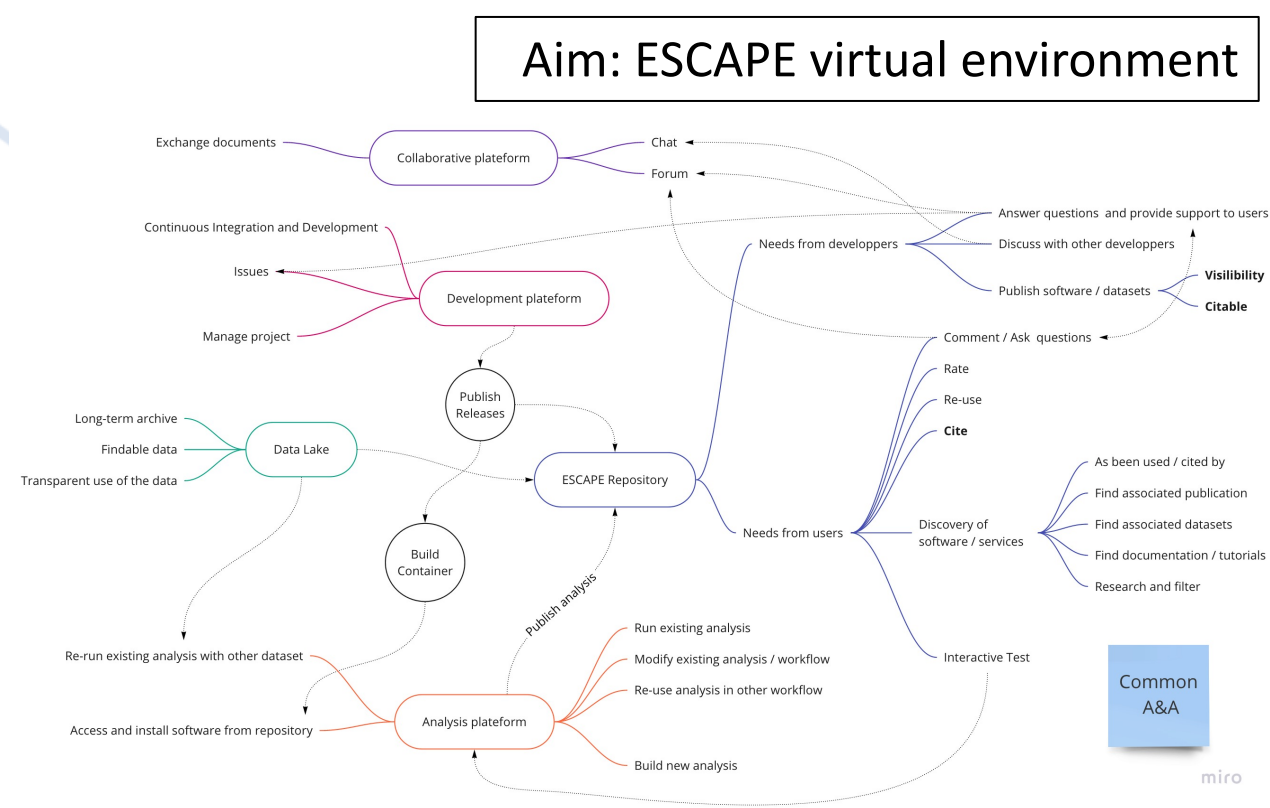
- Communities
- Views and downloads
- DOI: persistent identifier
- keywords
- Funding
- License
- Versions

- Share and cite



Conclusion

- 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/>



Appendix



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?



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>

license

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

Cited by

Link to project and article

Cite as

December 18, 2018

Software Open Access

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

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. Aleksić; 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

<ul style="list-style-type: none"> .gitignore 1_data.ipynb 2_results.ipynb 3_systematics.ipynb 4_naima.ipynb 5_crab_pulsar_nebula_sed.ipynb Dockerfile LICENSE README.md analysis.md binder <ul style="list-style-type: none"> environment.yml data <ul style="list-style-type: none"> README.md fact <ul style="list-style-type: none"> 20131103_103_dl3.fits 20131103_104_dl3.fits 20131103_105_dl3.fits 	<ul style="list-style-type: none"> 1.2 kB 3.0 kB 6.0 kB 100.8 kB 72.0 kB 65.4 kB 604 Bytes 1.5 kB 4.3 kB 2.9 kB 443 Bytes 512 Bytes 25.9 kB 25.9 kB 23.0 kB
---	--

Source code and data

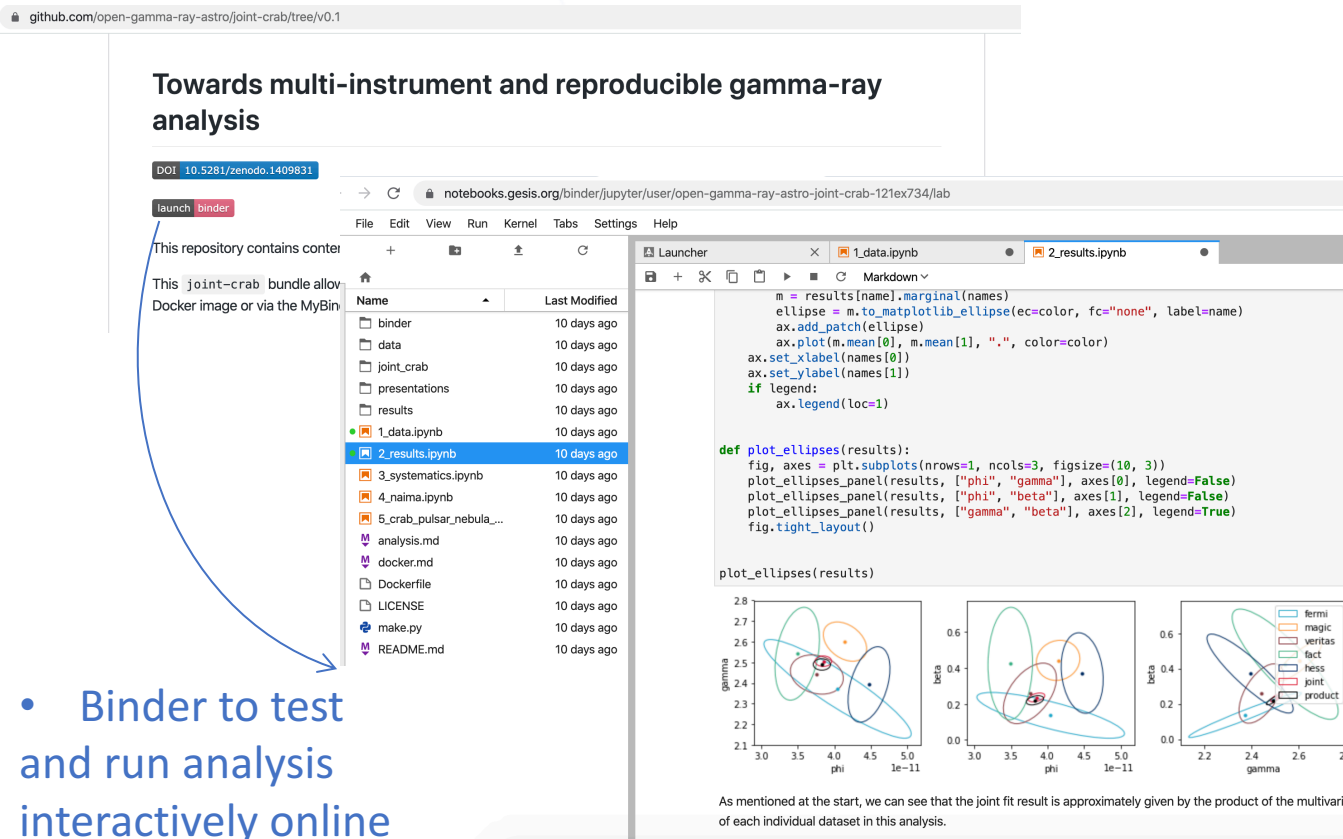


An example of project : The Crab bundle

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>



Towards multi-instrument and reproducible gamma-ray analysis

DOI: 10.5281/zenodo.1409831

launch binder

This repository contains container images for the joint-crab bundle. This joint-crab bundle allows Docker image or via the MyBinder.

Name	Last Modified
binder	10 days ago
data	10 days ago
joint_crab	10 days ago
presentations	10 days ago
results	10 days ago
1_data.ipynb	10 days ago
2_results.ipynb	10 days ago
3_systematics.ipynb	10 days ago
4_naima.ipynb	10 days ago
5_crab_pulsar_nebula_...	10 days ago
analysis.md	10 days ago
docker.md	10 days ago
Dockerfile	10 days ago
LICENSE	10 days ago
make.py	10 days ago
README.md	10 days ago

notebooks.gesis.org/binder/jupyter/user/open-gamma-ray-astro-joint-crab-121ex734/lab

```

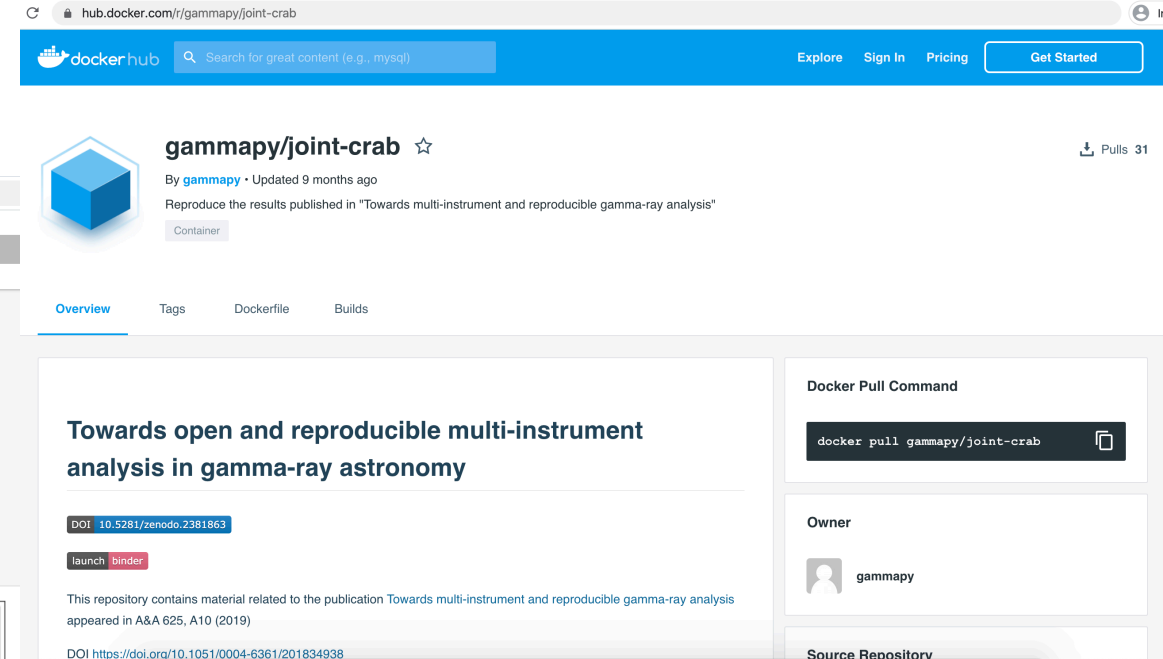
m = results[name].marginal(names)
ellipse = m.to_matplotlib_ellipse(ec=ec, fc="none", label=name)
ax.add_patch(ellipse)
ax.plot(m.mean[0], m.mean[1], ".", color=ec)
ax.set_xlabel(names[0])
ax.set_ylabel(names[1])
if legend:
    ax.legend(loc=1)

def plot_ellipses(results):
    fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(10, 3))
    plot_ellipses_panel(results, ["phi", "gamma"], axes[0], legend=False)
    plot_ellipses_panel(results, ["phi", "beta"], axes[1], legend=False)
    plot_ellipses_panel(results, ["gamma", "beta"], axes[2], legend=True)
    fig.tight_layout()

plot_ellipses(results)

```

As mentioned at the start, we can see that the joint fit result is approximately given by the product of the multivariate normal approximation for the of each individual dataset in this analysis.



hub.docker.com/r/gammapy/joint-crab

gammapy/joint-crab ☆

By gammapy · Updated 9 months ago

Reproduce the results published in "Towards multi-instrument and reproducible gamma-ray analysis"

Container

Overview Tags Dockerfile Builds

Towards open and reproducible multi-instrument analysis in gamma-ray astronomy

DOI: 10.5281/zenodo.2381863

launch binder

This repository contains material related to the publication Towards multi-instrument and reproducible gamma-ray analysis appeared in A&A 625, A10 (2019)

DOI: <https://doi.org/10.1051/0004-6361/201834938>

Docker Pull Command

```
docker pull gammapy/joint-crab
```

Owner

gammapy

Source Repository

- Docker to ensure reproducibility and ease of use

