ESCAPE
European Science Cluster of Astronomy & Particle physics ESFRI research Infrastructures

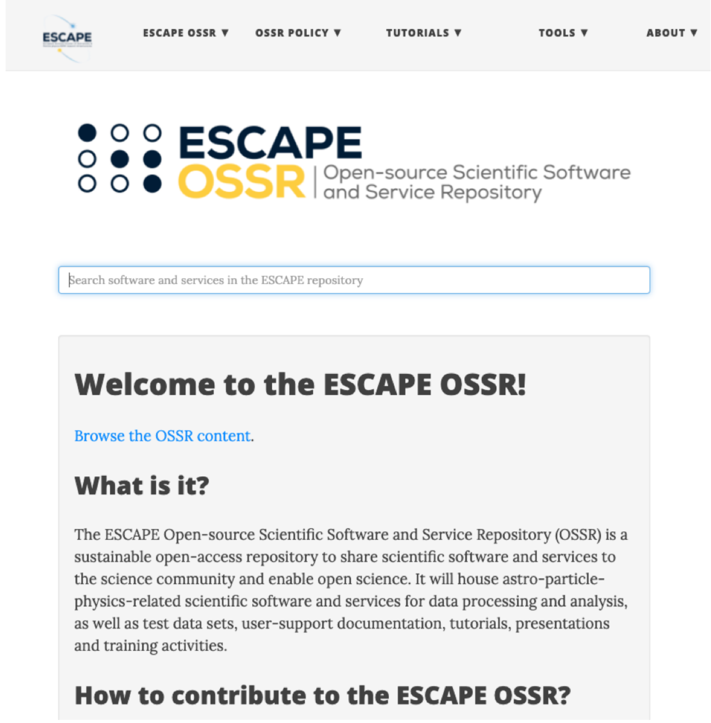
# ESCAPE OSSR

Thomas Vuillaume,
WP3-WP5 meeting, 2021-09-15

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.

# ESCAPE OSSR

- Portal = OSSR entry point: [http://purl.org/escape/ossr](http://purl.org/escape/ossr)
  - Search and browse the OSSR content
  - Software accessible by ESFRIs
  - OSSR policy: guidelines to contribute software or service to the OSSR
  - onboarding catalogue (to see what's coming soon into the OSSR)
  - tools: *eossr* library (see just after)
  - tutorials: contribute to the OSSR, how to use the CI, how to build containers


ESCAPE
ESCAPE OSSR ▾
OSSR POLICY ▾
TUTORIALS ▾
TOOLS ▾
ABOUT ▾
ESCAPE OSSR | Open-source Scientific Software and Service Repository
Search software and services in the ESCAPE repository

**Welcome to the ESCAPE OSSR!**

Browse the OSSR content.

**What is it?**

The ESCAPE Open-source Scientific Software and Service Repository (OSSR) is a sustainable open-access repository to share scientific software and services to the science community and enable open science. It will house astro-particle-physics-related scientific software and services for data processing and analysis, as well as test data sets, user-support documentation, tutorials, presentations and training activities.

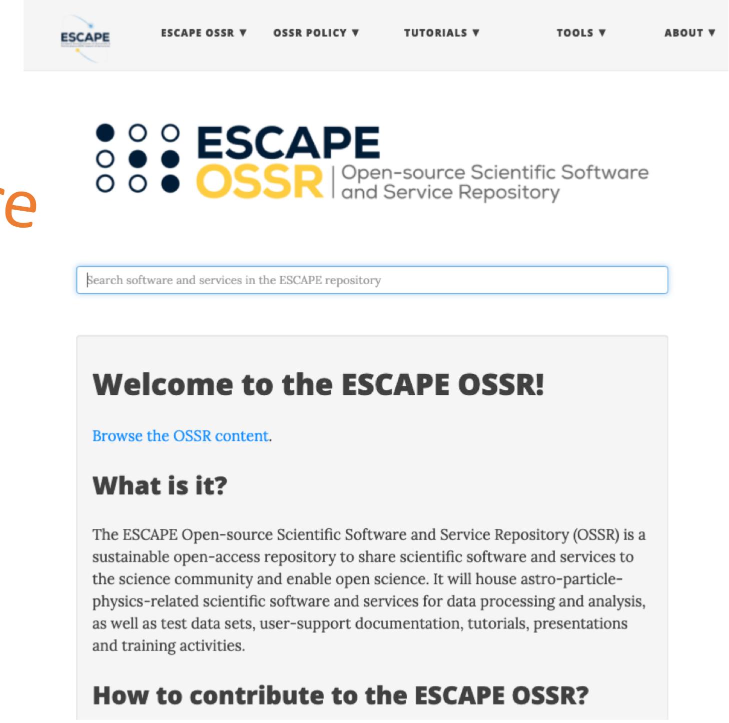
**How to contribute to the ESCAPE OSSR?**

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

# ESCAPE OSSR

You should start here

- Portal = OSSR entry point: http://purl.org/escape/ossr
  - Search and browse the OSSR content
  - Software accessible by ESFRIs
  - OSSR policy: guidelines to contribute software or service to the OSSR
  - onboarding catalogue (to see what's coming soon into the OSSR)
  - tools: *eossr* library (see just after)
  - tutorials: contribute to the OSSR, how to use the CI, how to build containers


ESCAPE
ESCAPE OSSR ▼
OSSR POLICY ▼
TUTORIALS ▼
TOOLS ▼
ABOUT ▼
ESCAPE OSSR Open-source Scientific Software and Service Repository
Search software and services in the ESCAPE repository

**Welcome to the ESCAPE OSSR!**

Browse the OSSR content.

**What is it?**

The ESCAPE Open-source Scientific Software and Service Repository (OSSR) is a sustainable open-access repository to share scientific software and services to the science community and enable open science. It will house astro-particle-physics-related scientific software and services for data processing and analysis, as well as test data sets, user-support documentation, tutorials, presentations and training activities.

**How to contribute to the ESCAPE OSSR?**

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

# EOSSR library

- Dev: https://gitlab.in2p3.fr/escape2020/wp3/eossr
- Doc: https://escape2020.pages.in2p3.fr/wp3/eossr/
  - regroup all current OSSR developments
  - Python
  - 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

- The discussion concerning these points should happen there, through issues (previous scattered issues have been moved here)

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

# OSSR API walkthrough

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

- `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 schema
  - implemented by adding codemeta.json file at the root of the project
- codemeta is a software metadata schema standard
- based on schema.org developed by major search engines (Google, Bing, Yahoo)
- it describes important information about software (license, purpose, authors, dependencies…) to facilitate discovery, adoption, and credit
- can be easily crosswalked to other metadata schemas if needed
- used by other services such as Software Heritage
- can be expanded/refined if needed

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

# Continuous Integration at your service

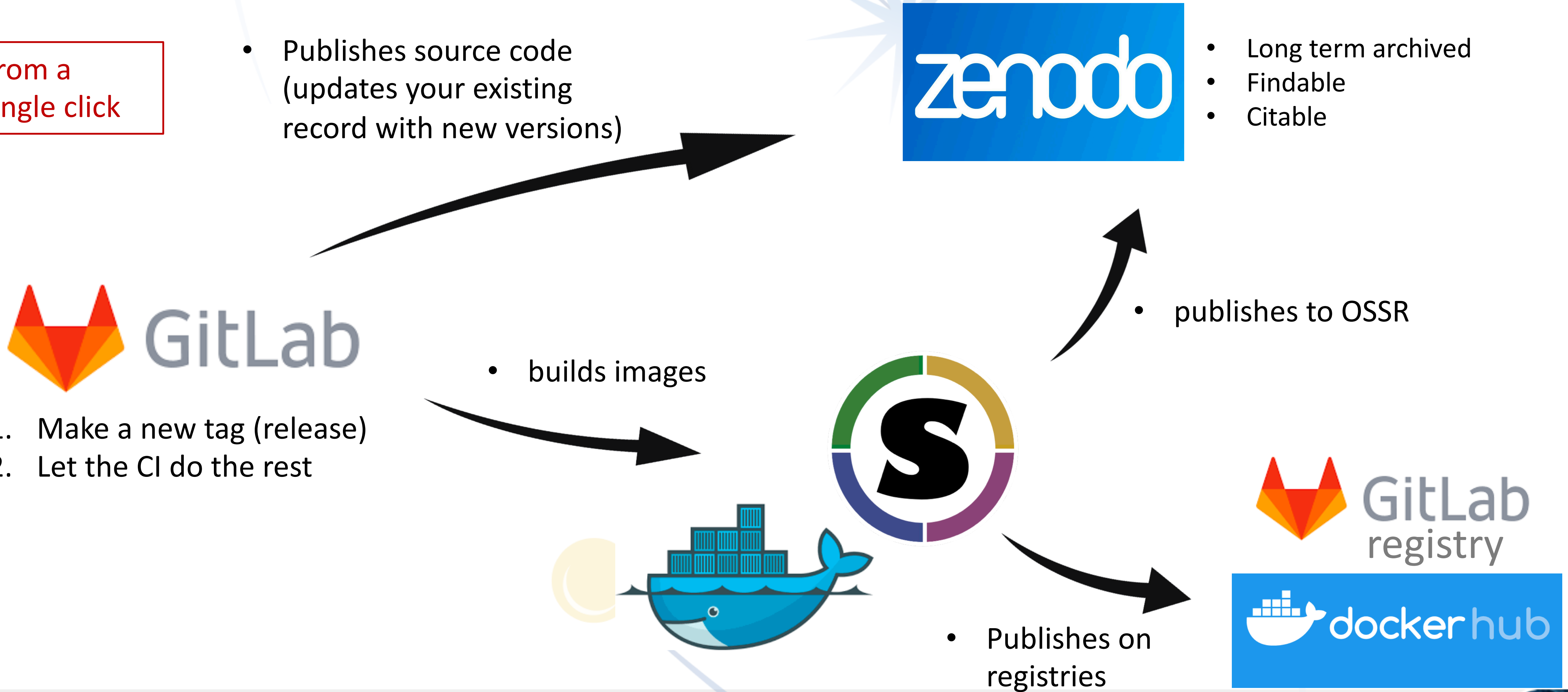
- based on codemeta.json in the code repository
- at the moment using GitLab CI (todo: same using github actions)
- upon software release:
  - update OSSR entry (using Zenodo API)
  - build a container
    - Singularity or Docker image → can be added into Zenodo entry
    - Docker containers → added 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

From a single click



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

- Long term archived
- Findable
- Citable

- publishes to OSSR

- builds images

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

- Publishes on registries

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

# Generate your CodeMeta file

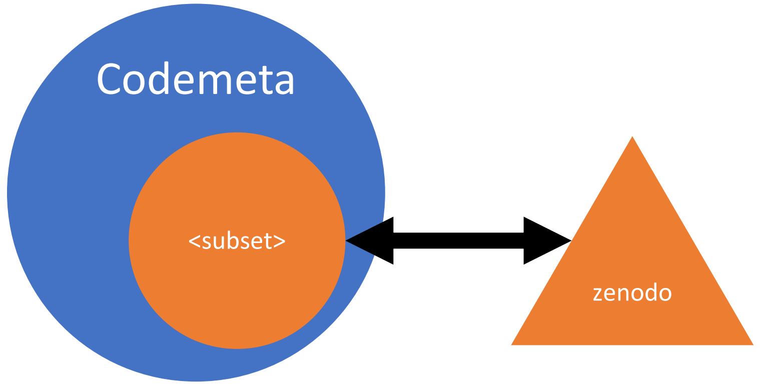
- codemeta.json file at the root of the project
- online generator: https://codemeta.github.io/codemeta-generator/
  - a similar custom escape codemeta generator could be built if we extand the schema. At the moment, use the official one.

- 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 an entry
  - the CI uses codemeta2zenodo (developed in WP3) to crosswalk codemeta.json into .zenodo.json

- 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: `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 ☺
  - extand schema?
  - *hack* the metadata, e.g. using keywords (*jupyter-notebook* has been suggested to manifest that the entry includes jupyter notebooks)

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

# Open questions

- How do you expect to get software?
  - Source code? docker / singularity / containers / images? Packages?
  - Install?
  - Docker-hub? Registry?

- Metadata
  - What kind of information you need for the ESAP?
    - Contains notebooks
    - Containers/Images:
      - Has one associated? --> if yes, URL
      - Is one? / Includes one in the record?
      - Docker/Singularity version?

    - Is workflow?

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