

GitHub Fundamentals. OpenAIRE & Zenodo infrastructures.

Technical annex for *D3.7 PoliRural GitHub Account* deliverable

Project	PoliRural	
Project title:	Future Oriented Collaborative Policy Development for Rural Areas and People	
Grant	818496	
Website:	www.polirural.eu	
Contact:	info@polirural.eu	
Version:	1.0	
Date:	September 2022	
Responsible:	TRAGSA	
Dissemination Level:	Public	X
	Confidential - only consortium members and European Commission Services	
Keywords:	GitHub, OpenAIRE, Open Science, Zenodo, Open Access, data sharing, open data, metadata.	

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 818496

Revision History

Revision no.	Date	Author	Organization	Description
0.x	March-August 2022		TRAGSA	Basic structure & successive drafts
1.0	September 2022		TRAGSA	Final version

Responsibility for the information and views set out in this publication lies entirely with the authors.

Every effort has been made to ensure that all statements and information contained herein are accurate, however the PoliRural Project Partners accept no liability for any error or omission.

Table of Contents

Table of Contents	3
List of Tables	4
List of Figures	5
Glossary	6
1 Introduction	8
2 GitHub Fundamentals	9
2.1 Git	9
2.2 Graphic User Interfaces (GUIs) for Git	10
2.3 GitHub and GitLab	11
2.4 GitHub functionalities	11
2.4.1 Key concepts	11
2.4.2 Key files in a repository	12
2.4.3 Security and privacy in GitHub	13
2.4.4 Repository Licensing in GitHub	14
2.4.5 GitHub-based project workflow	15
3 OpenAIRE	16
3.1 Zenodo	17
3.2 Zenodo Fair Principles	18
3.3 Zenodo interaction with GitHub	19
3.4 Other OpenAIRE Services	23
4 Conclusions	24
4.1 Advantages	24
4.2 Disadvantages	24
5 Annexes	25
5.1 Annex I: Version Control System platforms benchmarks	25
5.1.1 Main features comparative	25
5.1.2 Pricing scheme	26
5.1.3 Popularity	26
5.2 Annex II: Licensing	27
5.3 Annex III: Resources	28

List of Tables

TABLE 1 – DEFAULT AVAILABLE GITHUB LICENSES	15
TABLE 2 – OPENAIRE SERVICES (SOURCE).....	23
TABLE 3 – MAIN FEATURES AND MAIN PROVIDERS.....	25
TABLE 4 – GITHUB, GITLAB AND BITBUCKET PLANS AND PRICING.....	26

List of Figures

FIGURE 1 COMMON GIT OPERATIONS. VIA WIKIMEDIA COMMONS.	10
FIGURE 2 STRUCTURE OF A GITHUB-BASED PROJECT	15
FIGURE 3 OPENAIRE A.M.K.E. MEMBERS	16
FIGURE 4 ZENODO SECTION TO IMPORT GITHUB REPOSITORY.....	19
FIGURE 5 MANUALLY FILE UPLOAD TO ZENODO	20
FIGURE 6 ZENODO'S FILE TYPE SELECTOR	20
FIGURE 7 REQUIRED BASIC INFORMATION.....	20
FIGURE 8 ACCESS RIGHT AND LICENSE ARE BOTH REQUIRED METADATA.....	21
FIGURE 9 OTHER RECOMMENDED AND OPTIONAL METADATA AVAILABLE	21
FIGURE 10 FUNDING METADATA FIELDS IN EC / OPENAIRE PROJECTS	21
FIGURE 11 GITHUB VS. GITLAB VS. BITBUCKET GOOGLE TRENDS	26
FIGURE 12 EXAMPLE OF BASIC LICENSING SPECTRUM (ADAPTED FROM PERMISSIVE SOFTWARE LICENSE WIKI)	27
FIGURE 13 FLOSS-LICENSE-COMPATIBILITY (SOURCE: WIKIMEDIA COMMONS)	27
FIGURE 14 SOFTWARE LICENSE COMPATIBILITY FOR DERIVED WORKS (SOURCE: WIKIMEDIA COMMONS)	28

Glossary

- **Open Science** is an approach to the scientific process that focuses on spreading knowledge as soon as it is available using digital and collaborative technology. It is a policy priority for the European Commission and the standard method of working under its research and innovation funding programmes as it improves the quality, efficiency and responsiveness of research.
- **Open-Source Software (OSS)** refers to all that software whose original source code is made freely available and may be redistributed and modified. For other products other than software it is used generically **Open Source (OS)** when people can modify and share because its design is publicly accessible. It is common to see both concepts interchanged.
- **Open Access (OA)** refers to free access to information and unrestricted use of electronic resources for everyone.
- **FAIR Principles:** The FAIR principles (Findability, Accessibility, Interoperability and Reusability) describe how research outputs should be organised so they can be more easily accessed, understood, exchanged and reused.
- **European Open Science Cloud (EOSC)** is recognised by the *Council of the European Union* among the 20 actions of the policy agenda 2022-2024 of the *European Research Area (ERA)* with the specific objective to deepen open science practices in Europe. It aims to develop a 'Web of FAIR Data and services' for science in Europe upon which a wide range of value-added services can be built. These range from visualisation and analytics to long-term information preservation or the monitoring of the uptake of open science practices.
- **OpenAIRE:** OpenAIRE is a European project supporting *Open Science*. On the one hand, OpenAIRE is a network of dedicated Open Science experts promoting and providing training on Open Science. On the other, it is also a technical infrastructure harvesting research output from connected data providers.
- **Zenodo:** It is an all-purpose open research repository. It was created by OpenAire and CERN ("Conseil Européen pour la Recherche Nucléaire", or "or European Council for Nuclear Research") to provide a place for researchers to deposit publications, datasets and other research artefacts such as code, posters, presentations, etc.
- **DOI (Digital Object Identifier):** String of numbers, letters and symbols used to permanently and uniquely identify an article or document, and link to it on the web.
- **Licensing:** Formally *licensing* is the act of giving people official permission to do, have, or sell something. In the open source market, there exist around one hundred of licenses covering many use cases.
- **GDPR:** The General Data Protection Regulation (GDPR) is legislation that updated and unified data privacy laws across the European Union (EU). GDPR was approved by the European Parliament on April 14, 2016 and went into effect on May 25, 2018. GDPR replaces the EU Data Protection Directive of 1995.
- **Metadata:** A set of data that describes and gives information about other data. Metadata helps us understand the structure, nature, and context of the data. Metadata facilitates easy search and retrieval of data.
- **Version Control Systems (VCS):** A category of software tools able to record changes made to files by keeping track of modifications in the code.

- **Repository** (or **Repo**): Location where data, files and software packages are stored.
- **Git**: a free and open source distributed *version control system* designed to handle everything from small to very large projects with speed and efficiency.
- **GitHub**: A web-based hosting service for software development projects that use the Git revision control system plus access control, bug tracking, software feature requests, task management, continuous integration, and wikis for every project.
- **GUI**: A *graphical user interface* (GUI) is an interface through which, users interact with electronic devices such as computers and smartphones using icons, menus and other visual indicators or representations (graphics).
- **IDE**: An *integrated development environment* (IDE) is software for building applications that combines common developer tools into a single graphical user interface (GUI). Typically includes a source code editor, automatization, debugger and a consistent library of specialized plugins.
- **API**: An *application-programming interface* (API) is a set of protocols, routines, functions and/or commands used to facilitate interaction between distinct software services without having to know how they are implemented.
- **JSON** (*JavaScript Object Notation*) is a lightweight data-interchange format for storing and transporting data, easy for humans to read and write, programming language independent, and easy for machines to parse and generate. JSON is often used when data is sent from a server to a web page.

1 Introduction

This document compiles some basic information about GitHub and its interaction with the European OpenAIRE infrastructure, especially through Zenodo repository. It is not an in-deep analysis of those topics, but if more details were needed, some resources are provided along the text; regardless the report has been designed to be consistent without need of visiting the links.

As it happens with the whole modern Hardware and Software industry, everything can change to several levels and speeds, including relevant parts, workflows, prices, licensing, etc. Most of the links, documentation and references have been obtained according they were in the spring of 2022.

GitHub is the more popular software repository for Open Source projects, and in this document we will see it could be considered the natural choice to share not only code, if not also data, documents, schemes, graphics, etc.

In Chapter 2, we revise the fundamentals of GitHub, the main topics including a comparison with its direct competitor today, i.e. GitLab.

Chapter 3 is dedicated to the OpenAire and Zenodo infrastructure.

Benefits and drawbacks of using these methodologies are listed in Chapter 4.

Finally, three annexes have been added. The first one is a quick review of the current main *repository hosting* solutions, including features and pricing. The second one shows a visual synthesis of the Open Source licensing options. Finally, the last one offers a collection of interesting resources in the Internet.

2 GitHub Fundamentals

GitHub is primarily a collaborative *version control* build upon [git](https://git-scm.com/)¹, a free and open source ([GNU General Public License version 2](https://www.gnu.org/licenses/gpl-2.0.html))² *distributed version control* system.

The central functionality of these kind of tools is managing changes during software developing processes, but it also has some *side* features who might be suitable for non-programmers:

- each project has access control by user/team
- collaborative capabilities
- task management tools
- issue tracking, not only bugs, but also ideas, feedback or tasks
- easy (semi) automatic and secure publication of webpages and wikis
- social network, communication and diffusion tools out of the box
- easy integration on the workflow through Graphic User Interfaces (GUIs)

In the context of *OpenAIRE* infrastructure (and specifically related to Zenodo's repositories) GitHub is seamlessly integrated in them, and therefore it is a *default* solution for many users of the infrastructure.

This chapter shows the fundamentals and the main current features of this tool.

2.1 Git

The description as shown in the [official documentation](https://git-scm.com/doc)³ is:

Git is a fast, scalable, distributed revision control system with an unusually rich command set that provides both high-level operations and full access to internals.

Git operates from the *command line*, can be run locally from any computer, or in other environment as Internet or in any network, and can be installed for the principal operating systems by visiting the [git Downloads page](https://git-scm.com/downloads).⁴ Moreover, Git is freely available on the internet. Git comes under the *GPL open source license*, which means that you may obtain its source code and modify it to meet your needs.

Basically, Git supervises a local project folder, registers all the changes, and eventually, export them to a remote repository (*Push*). Other developers can collaborate in the project by importing it (*Fork* or *Clone*), working along in some area, and proposing to include those

¹ <https://git-scm.com/>

² <https://opensource.org/licenses/GPL-2.0>

³ <https://git-scm.com/docs/git.html>

⁴ <https://git-scm.com/downloads>

changes via *Pull request*. In addition, different versions of the software, or prototypes, etc. can be placed in different *Branches*.

Below there is a scheme of the typical use of Git:

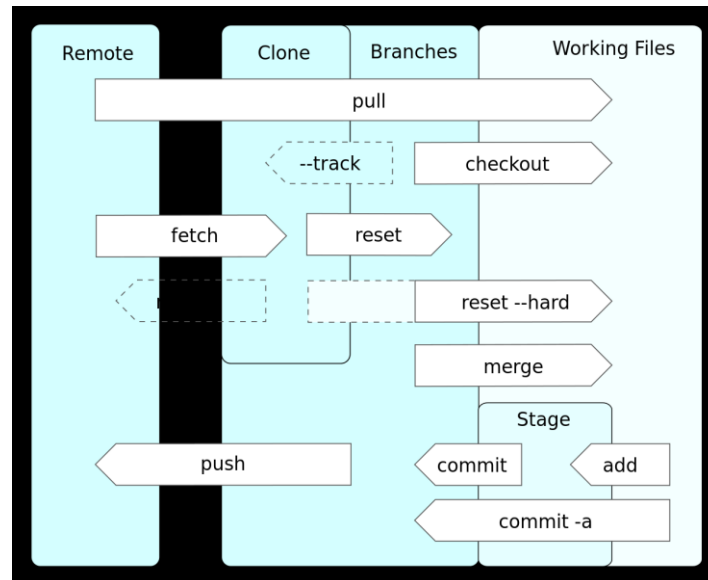


Figure 1 Common Git operations. Via Wikimedia Commons.⁵

Git is a powerful tool but have two critical limitations:

- Binary and big files. Git was designed to process mostly relatively small plain-text files. A *Git extension* is needed for overcome this limitation: [Git Large File Storage \(LFS\)](https://git-lfs.github.com/).⁶
- Git rely on commands, in general making the workflow not friendly for non-programmers roles.

2.2 Graphic User Interfaces (GUIs) for Git

Git itself comes with a couple of GUI tools for committing ([git-gui](https://git-scm.com/docs/git-gui)⁷) and browsing ([gitk](https://git-scm.com/docs/gitk)⁸) repositories. Being Git released under an *open source license* ([GNU GPLv2](https://opensource.org/licenses/GPL-2.0)⁹), many enterprises, both *open source* and *private*, have been designing and maintaining graphic interfaces offering to users a more visual and intuitive use of Git.

⁵ https://commons.wikimedia.org/wiki/File:Git_operations.svg

⁶ <https://git-lfs.github.com/>

⁷ <https://git-scm.com/docs/git-gui>

⁸ <https://git-scm.com/docs/gitk>

⁹ <https://opensource.org/licenses/GPL-2.0>

Some popular *proprietary solutions* are for example GitKraken, Sublime Merge, Sourcetree, etc.

In the *Open Source* arena, the most popular solution is [GitHub Desktop](#)¹⁰.

Moreover, most of the *source-code editors* and *Integrated Development Environment* (IDEs) support Git (and other) version control tools.

The official [Git site lists the most popular GUIs](#)¹¹ shortened by operating system. Additionally, Wikipedia offers an extensive list of [desktop Git GUIs](#)¹² too.

2.3 GitHub and GitLab

GitHub and Gitlab are today the main providers of native *cloud hosting* repositories with embedded *version control* through Git. Both are very similar in their *functionalities*, *pricing scheme* and *licensing*, and both services overcome (at least partially) the limitations of Git.¹³

Moreover, both platforms add other *key features* for modern developers (*CI/CD*, *DevOps*, *bug tracking*...), and other multidisciplinary tools (project management, wiki edition, web developing and secure hosting, social network...) suited for other roles too, like managers, data scientists, web developers, corporative communication, book edition, resources storage, professional networking, etc.

The choice of one or the other should be taken on a case-by-case basis; nevertheless, in the scenario of the interaction with *OpenAIRE / Zenodo*, GitHub emerges as a natural choice given it is already integrated in Zenodo.

In case that GitLab is needed, no major problems arise, and publisher could choose between manually upload the files through the Zenodo platform, or even import the GitLab repository to GitHub and then proceed with the import GitHub repository tool from Zenodo.

As a fact, most of the said in the next chapter (2.4) about GitHub could be said about GitLab too.

2.4 GitHub functionalities

2.4.1 Key concepts

The following is a collection of the key concepts to understand GitHub processes:

¹⁰ <https://desktop.github.com/>

¹¹ <https://git-scm.com/downloads/guis/>

¹² https://en.wikipedia.org/wiki/Comparison_of_Git_GUIs

¹³ For a complete comparative of both sites (updated to 2022), please visit: <https://slashdot.org/software/comparison/GitHub-vs-GitLab/>

- **Repository (or Repo).** - A central file storage location supervised by a *version control* to store multiple versions of files. Can be configured on a local machine for a single folder, on a server, or in the cloud. Users can create two kinds of repositories in *GitHub*:
 - *Public repositories.* - Anyone can access to them and use the content (according to the *license* provided in the repository).
 - *Private repositories.* - Only *owner / team members* can interact with it.
- **Commit.** - Wherever any file changes in the repository user can *commit* that change to the *local* or *remote* repo. After that, repositories could be updated in both directions:
 - *Pull.* - Transfer *commits* made or stored in a remote repository to a local repository.
 - *Push.* - Transfers *commits* made or stored locally to a remote repository.
- **Fetch.** - It adds changes from a remote repository to the local working branch without committing them.
- **Branch.** - A *branch* allow to work isolated from the current or production files. If the changes done are working, the user can send a *Pull request* to the owner and collaborators who might then *merge* the changes with the *main branch*.
- **Fork.** - Anyone with access to a repository can *fork* it, i.e. to make a copy of the original repository in *GitHub* (and then locally if needed) so the user can work in it, and then submit to the owner the changes via *Pull request*. If changes are approved the owner could integrate them in the *upstream branch* of the project.
- **Clone.** - Similar to *fork*, it makes a copy of the original repository on a local machine. In this way collaborators can be easily synchronize changes in both local and remote repositories.
- **Gist.** - It can be considered as a *one-file repository*, may be code fragments, or a configuration file, or a data table or even a to-do list. For the rest, it has the main features of an ordinary repository (integrated version control, forking and cloning, public and secret Gists, similar workflows, etc.).
- **Markdown.** - Though not mandatory, *Markdown* is the default syntax in *GitHub* platform. Details can be checked out in the *docs page*: [About writing and formatting on GitHub](https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/about-writing-and-formatting-on-github)¹⁴.
- **GitHub Marketplace.** - Is easy to extend *GitHub functionalities* via [all kind of apps](https://github.com/marketplace)¹⁵.

For more basic information, *GitHub Docs* offers a full [GitHub glossary](https://docs.github.com/en/get-started/quickstart/github-glossary)¹⁶.

2.4.2 Key files in a repository

Most of the repositories contain some singular files user should know about, being of particular importance:

¹⁴ <https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/about-writing-and-formatting-on-github>

¹⁵ <https://github.com/marketplace>

¹⁶ <https://docs.github.com/en/get-started/quickstart/github-glossary>

- **README.** - The file README.md (if markdown format is used, though any plain text is admitted, e.g. README.txt –The same happens with most of the files below), is the first information that visitors to the repository can see. Therefore it is used often to explain the project, show its degree of maturity, point to documentation files or folders, or whatever information that the owner may find relevant for viewers or users. Moreover, a README file can be added to your personal/Organization's profile, becoming in a front page for the project, product or enterprise.
- **LICENSE.** - The LICENSE file shows the repository legal license, an important piece of information when the repository is public and some rights are needed to be preserved, or when your project contains other peoples' work with its own license. In the *chapter 2.4.4* and in the *Annex II* some more information is provided.
- **CITATION.cff.** - It explains how authors would like people to cite their project. Though it is also a plain text file, some format to be machine-readable is needed. See details in the GitHub Docs: [About CITATION files](#).¹⁷
- **.gitignore.** - This file is very useful when the user is working locally, contributing to the project via pull-requests, but need maintaining certain files and / or folders out of the version control, i.e. ignore them by not including them in the GitHub repository. An example can be seen in *chapter 2.4.3*.
- **CODE_OF_CONDUCT.** - When the repository is community oriented and many people interact in the project, a clear text about the expected participants' behaviour may be central.

Depending on the complexity of the project, or the number of participants, etc. other files might be also important: **AUTHORS, CHANGELOG, CONTRIBUTING, ACKNOWLEDGMENTS,** etc.

2.4.3 Security and privacy in GitHub

GitHub has a good privacy (including EU's GDPR), security, audit, and compliance policies. Extensive documentation about this dimension of the offered services can be found in the [GitHub Security page](#).¹⁸

¹⁷ <https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-citation-files>

¹⁸ <https://github.com/security>

Other layer of security can be added by configuring [SSH protocol](#)¹⁹, which allow authenticate users and grant them access to a given (private or public) repository. This option should be used in cases where sensitive data are involved.

In terms of privacy, the human factor is central because the major vulnerabilities come from misused management of files and/or sensitive data, something that escalates fast when many people have access to one or more public repositories. Git repositories have a method to avoid that certain files and folders may be end supervised by the control version and/or, subsequently, being shared to remote repositories. It consists in simply excluding files, folders or certain patterns using a single plain-text file (*.gitignore*) which lists all the folders/files to be excluded.

For example:

```
## Word temporary files
*.tmp
~$*.doc*

## Private files/folders
Aux*.*

/notshared/
/Docs/embargoed

## gitignore itself
.gitignore
```

For extra information about *.gitignore*, please visit for example the [git documentation](#)²⁰.

A collection of *.gitignore* templates can be consulted in this repository²¹.

2.4.4 Repository Licensing in GitHub

Although the owner of the repository can opt for any license, Github includes automatically the appropriate legal text for a number of popular licenses, and by default, assigns the authorship to the repository owner.

A typical method to classify licenses is related to how permissive is the software (or other digital products), and in this fashion is presented in the following table.

¹⁹ <https://docs.github.com/en/github-ae@latest/authentication/connecting-to-github-with-ssh/about-ssh>

²⁰ <https://git-scm.com/docs/gitignore>

²¹ <https://github.com/github/gitignore>

Public Domain

- Creative Commons Zero v1.0 Universal
- The Unlicense

Permissive	Less permissive
• Apache License 2.0	• GNU General Public License v3.0
• MIT License	• GNU Affero General Public License v3.0
• BSD 2-Clause "Simplified" License	• GNU General Public License v2.0
• BSD 3-Clause "New" or "Revised" License	• GNU Lesser General Public License v2.1
• Boost Software License 1.0	• Eclipse Public License 2.0
• Mozilla Public License 2.0	

Table 1 – Default available GitHub licenses

The task of choosing the correct license is not always easy, GitHub maintain a site that may help to pick the right one: [Choose an open source license | Choose a License](#)²²

A quick review about the state of the art on Open Source licensing can be checked in *Annex II: Licensing*.

2.4.5 GitHub-based project workflow

The use of GitHub in a project is in some way a *paradigm change* because several actors, environments and roles could be integrated in the workflow, including users and communities, publishers, reviewers, general public, etc. The report “*Ten Simple Rules for Taking Advantage of Git and GitHub*”²³ - CC BY 4.0²⁴ displays a sample of working with GitHub.

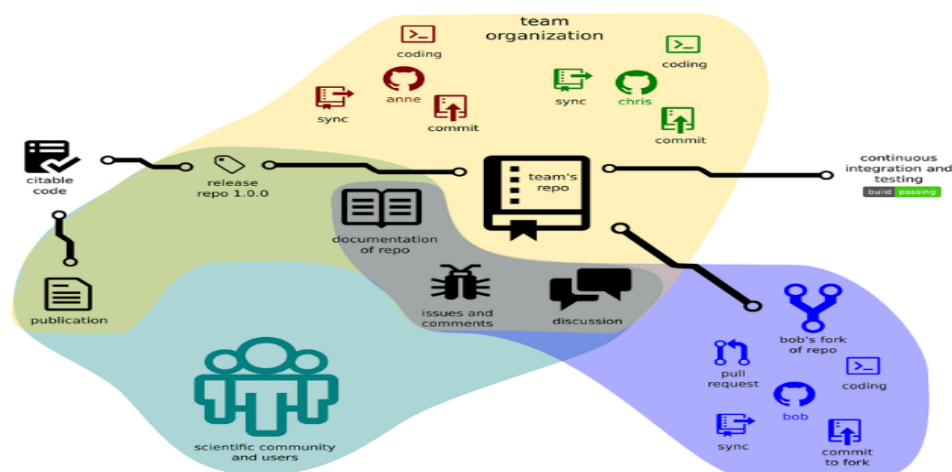


Figure 2 Structure of a GitHub-based project

²² <https://choosealicense.com/>

²³ <https://doi.org/10.1371/journal.pcbi.1004947>

²⁴ <https://creativecommons.org/licenses/by/4.0/>

3 OpenAIRE

In the site [About page](#)²⁵ we can read:

OpenAIRE is a Non-Profit Partnership, established in 2018 as a legal entity, OpenAIRE A.M.K.E, to ensure a permanent open scholarly communication infrastructure to support European research.

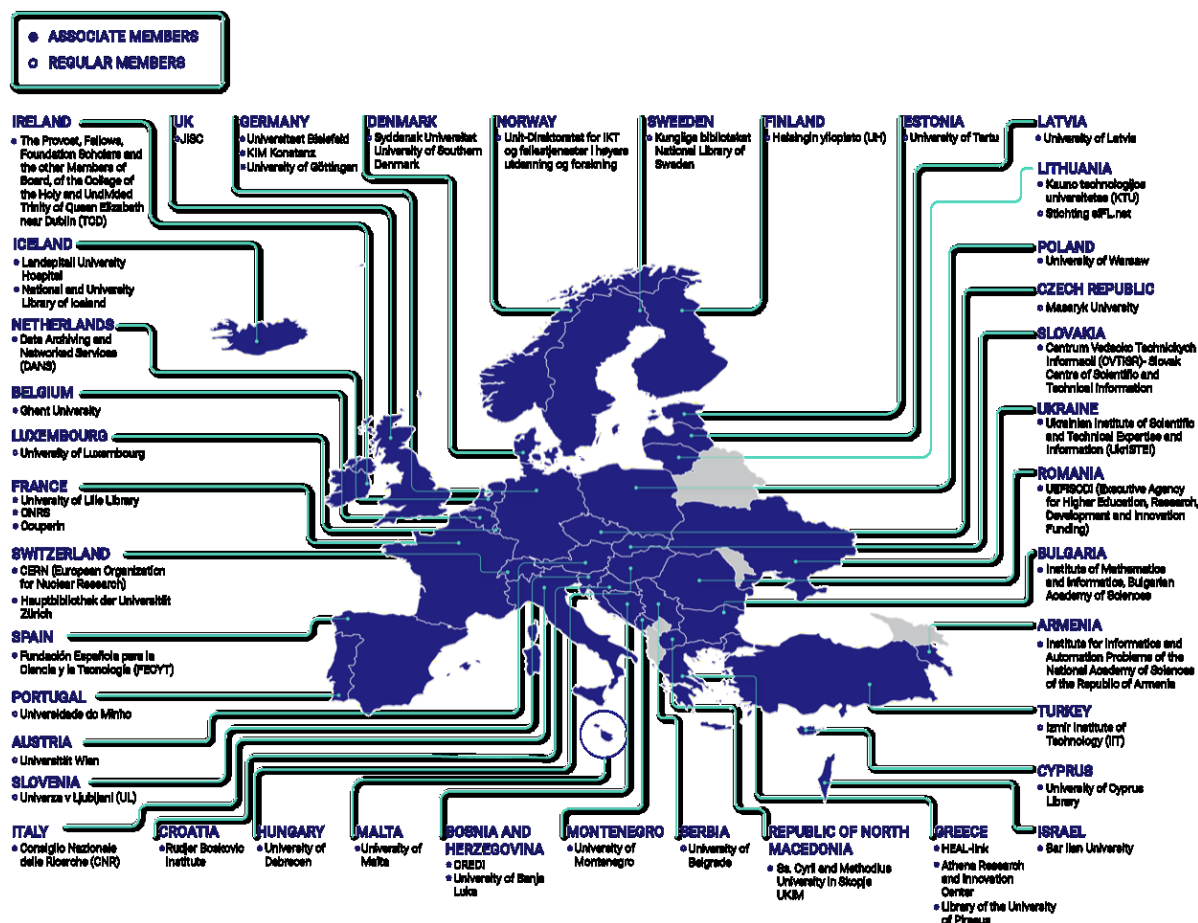


Figure 3 OpenAIRE A.M.K.E. members

[OpenAIRE mission](#)²⁶ is described as:

OpenAIRE's mission is closely linked to the mission of the European Commission: to provide unlimited, barrier free, open access to research outputs financed by public funding in Europe. OpenAIRE fulfils the EOSC vision substantially, as its operations already provide the glue for many of the user and research driven functionalities, whether these come from the long tail of science (repositories and local support) or domain disciplined research communities or Research Infrastructures.

²⁵ <https://www.openaire.eu/about>

²⁶ <https://www.openaire.eu/mission-and-vision>

The [European Open Science Cloud \(EOSC\)](#)²⁷ indeed share with OpenAIRE objectives as seamless access, FAIR (Findability, Accessibility, Interoperability and Reusability) management, and reliable reuse of research data and all other digital objects produced along the research life cycle (e.g. methods, software and publications).

OpenAIRE contributes actively through its large network, supporting Open Source policies, and helping via the [National Open Access Desks \(NOADs\)](#)²⁸. More about the **OpenAIRE's** contribution to **EOSC** can be checked in the paper: “Achieving Open Science in the European Open Science Cloud: Setting out OpenAIRE’s vision and contribution to EOSC” ([DOI | 10.5281/zenodo.3610132](#))²⁹.

3.1 Zenodo

Zenodo is an open data repository running with the *open source software* [Invenio](#)³⁰, created and maintained by CERN and enabling researchers to share and preserve any research outputs in any size, format, and from any science / field.

Given that Zenodo is part of the OpenScience.eu infrastructure, let see some excerpts from OpenScience.eu about Zenodo³¹:

Zenodo was launched within the frame of the OpenAIRE project, which was commissioned by the European Commission to provide open access to research outputs financed by public funding in Europe.

[...]

As an OpenAIRE Partner, the European Organization for Nuclear Research known as CERN created Zenodo as a “Catch-all” data repository service for EU researchers in 2013. Anyone with research output may deposit it (e.g. articles, datasets, images, posters, software).

[...]

Zenodo accepts large files (up to 50GB) without format restriction. The data is assigned a Digital Object Identifier (DOI), which means that you do not need to publish in a journal in order to obtain a unique and permanent DOI, which makes your publication more easily citable according to international standards.

²⁷ https://ec.europa.eu/info/research-and-innovation/strategy/strategy-2020-2024/our-digital-future/open-science/european-open-science-cloud-eosc_en

²⁸ <https://www.openaire.eu/openaire-and-eosc>

²⁹ <https://zenodo.org/record/3610132>

³⁰ <https://invenio-software.org/>

³¹ <https://openscience.eu/zenodo/>

3.2 Zenodo Fair Principles

Zenodo aligns with **FAIR Principles** definition as referenced from *Wilkinson, M. D. et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci. Data 3:160018 doi: [10.1038/sdata.2016.18](https://doi.org/10.1038/sdata.2016.18) (2016)*³².

The key policies in these principles as can be read in the [Zenodo Principles page](https://about.zenodo.org/principles/)³³ are listed below:

- **To be Findable.** - FAIR principles promote an extensive use of *metadata* following recommended terms of the [DataCite's Metadata Schema](https://schema.datacite.org/)³⁴, including *DOI* as a mandatory *metadata* field, being then recorded and indexed, and therefore immediately searchable directly in *Zenodo's search engine* after publishing.
- **To be Accessible.** - To be accessible, the metadata and data should be understandable to both humans and machines, and data should be stored in a trusted repository. Metadata for individual records and collections are harvestable by two methods:
 - using the [OAI-PMH protocol](https://zenodo.org/oai2d)³⁵ by the record identifier and the collection name
 - through the public [REST API](https://developers.zenodo.org/)³⁶
- **To be Interoperable.** - To be *interoperable*, metadata should use a formal, accessible, shared, and broadly applicable language for knowledge representation, such as agreed-upon controlled vocabularies. Zenodo uses [JSON Schema](https://json-schema.org/)³⁷ as internal representation of metadata, and is ready for export to other *international schemas* formats.
- **To be Reusable.** - In this aspect some features and policies are included:
 - License is one of the mandatory terms in Zenodo's metadata, and is referring to an [Open Definition](https://opendefinition.org/)³⁸ license (succinctly, "Open data and content can be freely used, modified, and shared by anyone for any purpose").
 - Data downloaded by the users is subject to the license specified in the metadata by the owner.
 - All data and metadata uploaded is traceable to a registered *Zenodo user*.

³² <https://doi.org/10.1038/sdata.2016.18>

³³ <https://about.zenodo.org/principles/>

³⁴ <https://schema.datacite.org/>

³⁵ <https://zenodo.org/oai2d>

³⁶ <https://developers.zenodo.org/>

³⁷ <https://json-schema.org/>

³⁸ <https://opendefinition.org/>

- Zenodo is not a domain-specific repository, still, through compliance with DataCite's Metadata Schema³⁹ meets one of the broadest cross-domain standards available.

Metadata data and metadata itself will be hosted for the lifetime of the repository. This is currently the lifetime of the host laboratory CERN, which has an experimental programme defined for the next 20 years at least. Metadata are stored in high-availability database servers at CERN, which are separate to the data itself.

3.3 Zenodo interaction with GitHub

GitHub is today the main platform for software developers (above 70 million users) and many important *Open Source* projects are hosted there, including the [European Commission projects](#)⁴⁰, or the [OpenAIRE](#)⁴¹ and [Zenodo](#)⁴² repositories.

In this scenario is perfectly natural that a repository as Zenodo do all possible for ease the compatibility with GitHub. For example, a user can easily sign up to Zenodo through GitHub accounts (or alternatively via ORCID)⁴³, and then easily import to Zenodo their public GitHub repositories.⁴⁴

Once the user/publisher is logged in, Zenodo can easily access to the [Github repository importer section](#)⁴⁵.

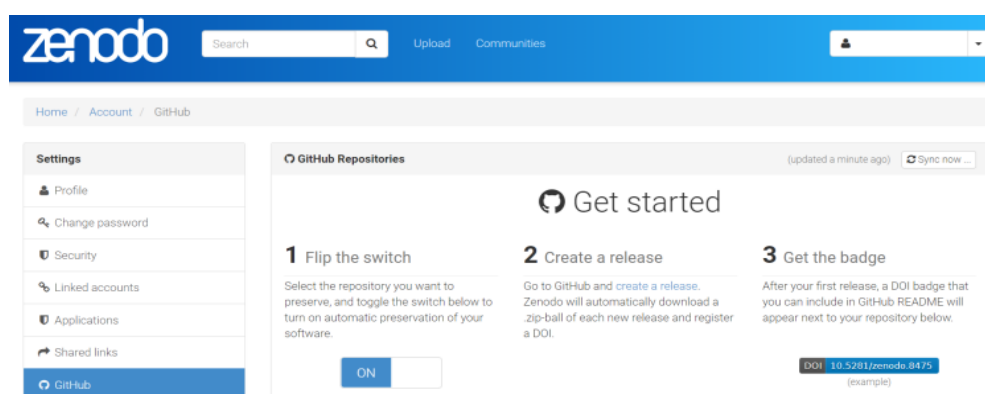


Figure 4 Zenodo section to import Github repository

³⁹ <https://schema.datacite.org/meta/kernel-4.4/>

⁴⁰ <https://github.com/ec-europa>

⁴¹ <https://github.com/openaire>

⁴² <https://github.com/zenodo>

⁴³ <https://www.zenodo.org/login/>

⁴⁴ <https://zenodo.org/account/settings/github/> - A Zenodo's account is needed

⁴⁵ <https://www.zenodo.org/account/settings/github/> - A Zenodo's account is needed

Alternatively, publisher can do it by [manually loading](#)⁴⁶ the pertinent files.

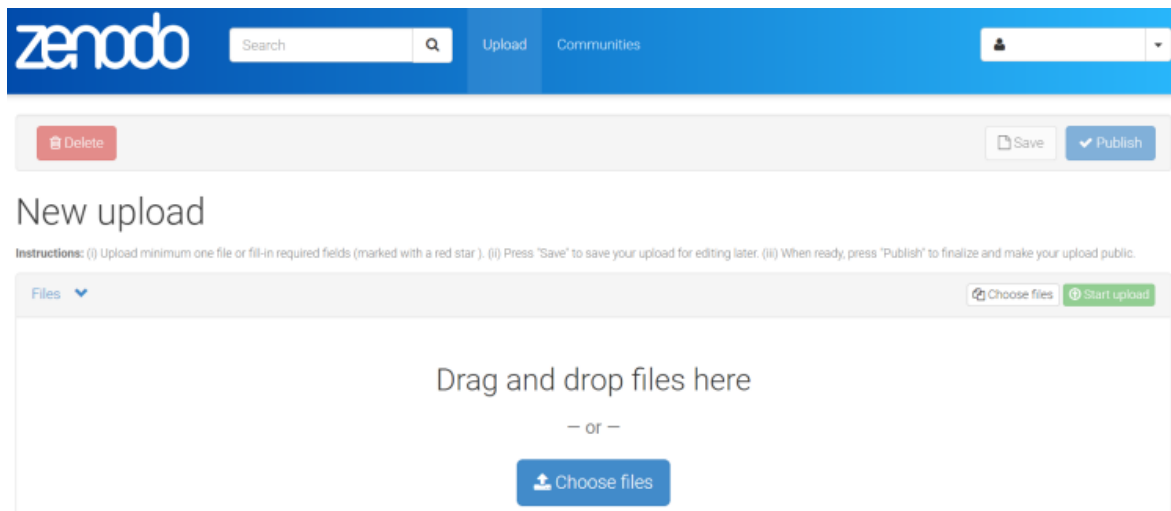


Figure 5 Manually file upload to Zenodo

In this case, the user must load manually metadata too, but Zenodo system ease the task using visual menus. Below some captures of the process can be seen.

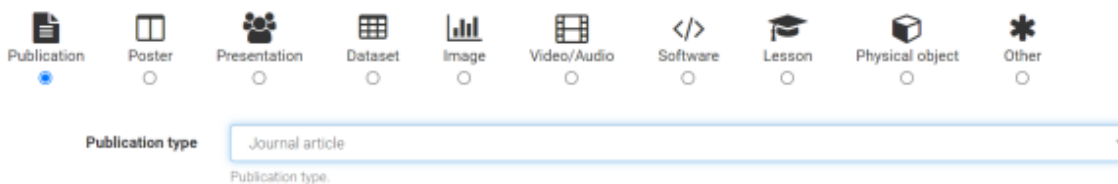


Figure 6 Zenodo's file type selector

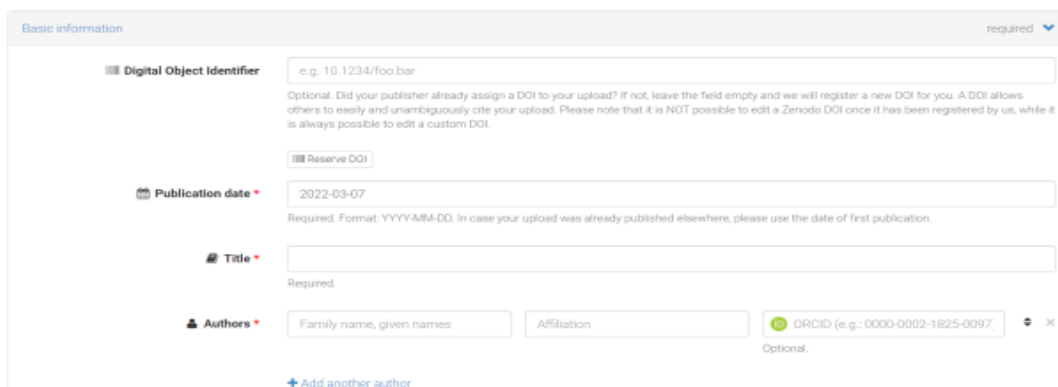


Figure 7 Required basic information

⁴⁶ <https://www.zenodo.org/deposit/new> - A Zenodo's account is needed

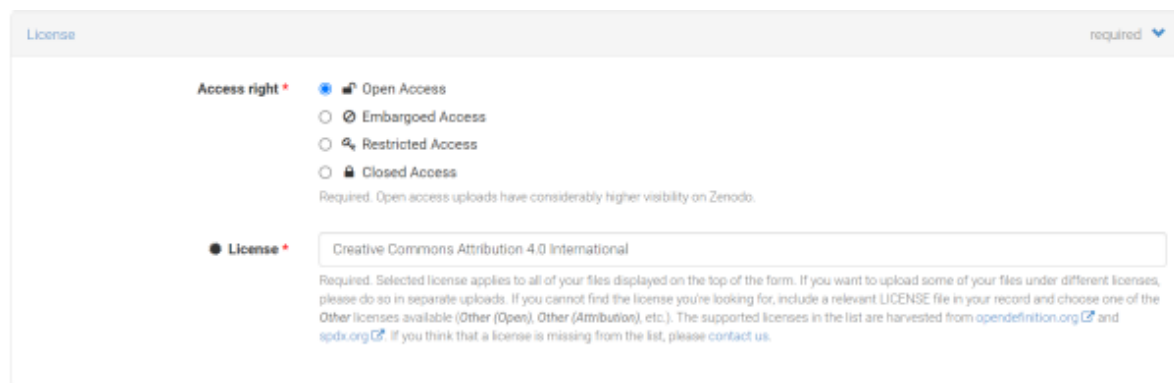


Figure 8 Access right and license are both required metadata

Aside of the mandatory metadata fields, Zenodo offers some recommended and optional fields, which may improve the documents findability in Zenodo and OpenAIRE search engines. Depending on the type of document, it might be advisable to populate as many fields as possible.



Figure 9 Other recommended and optional metadata available

For example, in the context of European Commission projects (through OpenAIRE), probably the *funding section* could be required even not being mandatory for Zenodo.

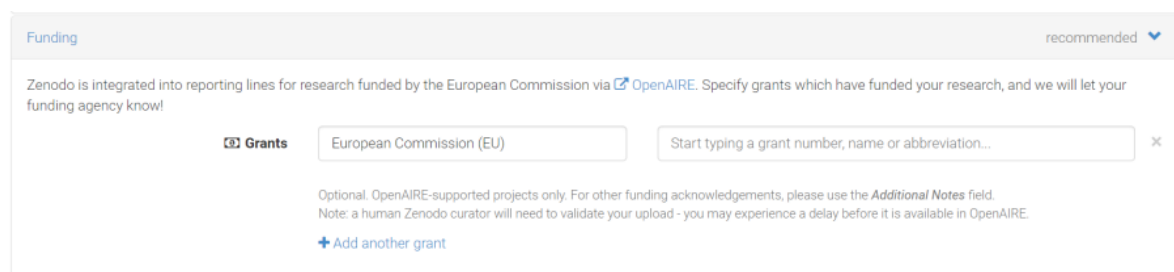


Figure 10 Funding metadata fields in EC / OpenAIRE projects

Metadata from GitHub are automatically extracted, but if some customization is needed, a file (*.zenodo.json*) must be included in the root of the repository.⁴⁷ It is a simple plain-text file easily human-readable and/or editable. Below an arbitrary example.

```
{
  "creators": [
    {
      "orcid": "0000-0002-1825-0097",
      "affiliation": "Feline research institute",
      "name": "Field, Gar"
    },
    {
      "orcid": "0000-0002-1825-0097",
      "affiliation": "Feline research institute",
      "name": "Cat, Felix"
    }
  ],
  "license": "Apache-2.0",
  "title": "Red-Dot: ML-powered laser vector detection",
  "related_identifiers": [
    {
      "scheme": "doi",
      "identifier": "10.1234/software.paper.5678",
      "relation": "isDocumentedBy",
      "resource_type": "publication-article"
    }
  ],
  "keywords": ["Cats", "Laser", "Behavior"],
  "communities": [
    {"identifier": "software-cats"}
  ],
  "grants": [{"id": "777541"}]
}
```

This functionality makes easy to manage the appropriate information and metadata in each repository to be linked with Zenodo. At the time of including the *.zenodo.json* file the [entities deposition documentation](https://developers.zenodo.org/#entities)⁴⁸ should be checked, and some tool for testing the file formatting should be used, e.g. [JSON Formatter & Validator](https://jsonformatter.curiousconcept.com/)⁴⁹.

⁴⁷ <https://developers.zenodo.org/#github>

⁴⁸ <https://developers.zenodo.org/#entities>

⁴⁹ <https://jsonformatter.curiousconcept.com/>

3.4 Other OpenAIRE Services

Zenodo is only a part of the OpenAIRE infrastructure, a list of other free services oriented to adopt easily Open Science, can be seen below:

Service	Motto/Description
<ul style="list-style-type: none"> • Argos: for administrators ⁵⁰ 	Bring your Data Management Plans closer to where data are generated, analysed and stored.
<ul style="list-style-type: none"> • OpenCitations ⁵¹ 	Independent not-for-profit infrastructure organization for open scholarship dedicated to the publication of open bibliographic and citation data by the use of Semantic Web (Linked Data) technologies.
<ul style="list-style-type: none"> • Research Graph ⁵² 	Open resource that aggregates a collection of research data properties (metadata, links) available within the OpenAIRE Open Science infrastructure for funders, organizations, researchers, research communities and publishers to interlink information by using a semantic graph database approach.
<ul style="list-style-type: none"> • Provide ⁵³ 	The Provide Dashboard is a one-stop-service where content providers interact with OpenAIRE and become a building block of a global Open Research community. A gateway to the European Open Science Cloud.
<ul style="list-style-type: none"> • Amnesia ⁵⁴ 	Perform research and share your results that satisfy GDPR guidelines by using data anonymization algorithms.
<ul style="list-style-type: none"> • Connect ⁵⁵ 	Build a Gateway to your community's open and linked research outcomes
<ul style="list-style-type: none"> • UsageCounts ⁵⁶ 	Collecting usage data from Open Science content providers repositories, journals, and other scientific data sources.
<ul style="list-style-type: none"> • Monitor ⁵⁷ 	Work together with us to construct indicators that suit your needs and to view, understand and visualize research statistics and indicators.
<ul style="list-style-type: none"> • Zenodo ⁵⁸ 	Open, dependable repository for all of scholarship, enabling researchers from all disciplines to share and preserve their research outputs, regardless of size or format. Free to upload and free to access, Zenodo makes scientific outputs of all kinds citable, shareable and discoverable for the long term.

Table 2 – OpenAIRE services ([source](https://www.openaire.eu/factsheets) ⁵⁹)

⁵⁰ <https://argos.openaire.eu/>

⁵¹ <https://opencitations.net/>

⁵² <https://graph.openaire.eu/>

⁵³ <https://provide.openaire.eu/>

⁵⁴ <https://amnesia.openaire.eu/>

⁵⁵ <https://connect.openaire.eu/>

⁵⁶ <https://usagecounts.openaire.eu/>

⁵⁷ <https://monitor.openaire.eu/>

⁵⁸ <https://www.zenodo.org/>

⁵⁹ <https://www.openaire.eu/factsheets>

4 Conclusions

The use of GitHub and Zenodo in the context of OpenAIRE grants the European Open Science standards and policies respect to research outputs. Its application in current (or future) projects might presents pros and cons that can be summarized in the below points.

4.1 Advantages

- Free DOIs assignment for public records and automatic indexing in OpenAIRE
- Automatic versioning of code, datasets, documents, reports, etc...
- Automatic citations
- Better and easier content diffusion
- Repository usage statistics
- Security and privacy standards compliant
- 20 years of persistence of data and metadata in the CERN Data Center

4.2 Disadvantages

- It is necessary to face some paradigm changes in the workflow, and the learning curve could be stepped for certain roles
- Extra workload for developers, authors and managers
- Repositories and therefore contents get exposed to the Internet in the wild
- Need for some kind of maintenance of the repositories
- Choosing licenses may be conflictive and time consuming
- Extra caution for [GDPR compliance](https://gdpr.eu/)⁶⁰.

⁶⁰ <https://gdpr.eu/>

5 Annexes

5.1 Annex I: Version Control System platforms benchmarks

The relevant info from the three more popular solutions today in the area of code hosting and version control using *Git* are **GitHub**, **GitLab** and **BitBucket**.

5.1.1 Main features comparative






















Feaures	GitHub	GitLab	BitBucket
Number of users	73 million	30 million	10 million
Free Private Repositories			
Free Public Repositories			
Navigation Usability			
Graphical User Interface (GUI)	 Free and Open Source GitHub Desktop ⁶¹		 Free but Proprietary Sourcetree ⁶²
Large file storage			
Open Source			
3 rd party tools Integration			
Hosting static sites	GitHub Pages ⁶³	GitLab Pages ⁶⁴	Bitbucket Cloud ⁶⁵ (with limitations), or via commercial plugin: Pages for Bitbucket Server ⁶⁶

Table 3 – Main features and main providers (2022)

⁶¹ <https://desktop.github.com/>

⁶² <https://www.sourcetreeapp.com/>

⁶³ <https://pages.github.com/>

⁶⁴ <https://about.gitlab.com/stages-devops-lifecycle/pages/>

⁶⁵ <https://support.atlassian.com/bitbucket-cloud/docs/publishing-a-website-on-bitbucket-cloud/>

⁶⁶ <https://marketplace.atlassian.com/apps/1212525/pages-for-bitbucket-server>

5.1.2 Pricing scheme

Current plans - Spring 2022




Plans	GitHub	GitLab	BitBucket
Free			
Medium	<i>Team</i> \$44 per user/year	<i>Premium</i> \$19 per user/month	<i>Standard</i> \$3 per user/month Starts \$15 /month
Top	<i>Enterprise</i> \$231 per user/year	<i>Ultimate</i> \$99 per user/month	<i>Premium</i> \$6 per user/month Starts \$30 /month
Self Hosted / Self Managed	<i>Enterprise Server</i> \$231 per user/year	<i>For all tiers</i> Self-Managed Feature Comparison ⁶⁷	<i>Data Center</i> \$2300 per 25 users/year
Details	Pricing - Plans for every developer ⁶⁸	GitLab Pricing ⁶⁹	Bitbucket - Pricing ⁷⁰

Table 4 – GitHub, GitLab and BitBucket plans and pricing

5.1.3 Popularity

Google trends displays the interest or popularity using as metric the number of searches:



Figure 11 GitHub vs. GitLab vs. BitBucket [Google Trends](#)⁷¹

⁶⁷ <https://about.gitlab.com/pricing/self-managed/feature-comparison/>

⁶⁸ <https://github.com/pricing>

⁶⁹ <https://about.gitlab.com/pricing/>

⁷⁰ <https://www.atlassian.com/software/bitbucket/pricing>

⁷¹ <https://trends.google.es/trends/explore?cat=13&date=all&q=GitHub,GitLab,BitBucket>

5.2 Annex II: Licensing

Currently there are about one hundred open source licenses in the market, and choosing one is not easy, actually may become a critical task that might compromise the whole project. As far as an in deep analysis of the process of licensing is out of the scope of this document, this annexe is designed to get a quick view of the current state of the art in the field.

This licensing spectrum focused to proprietization is explained in the following image:

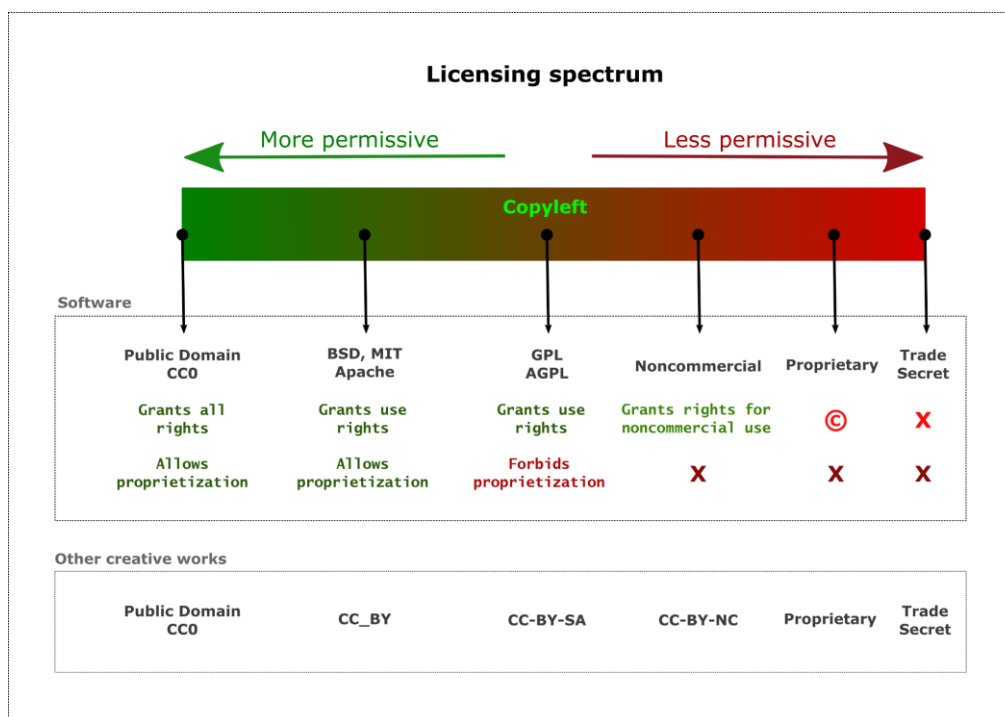


Figure 12 Example of basic licensing spectrum (Adapted from [Permissive software license wiki](https://en.wikipedia.org/wiki/Permissive_software_license)⁷²)

Example of free and open-source software (FOSS) license compatibility schema:

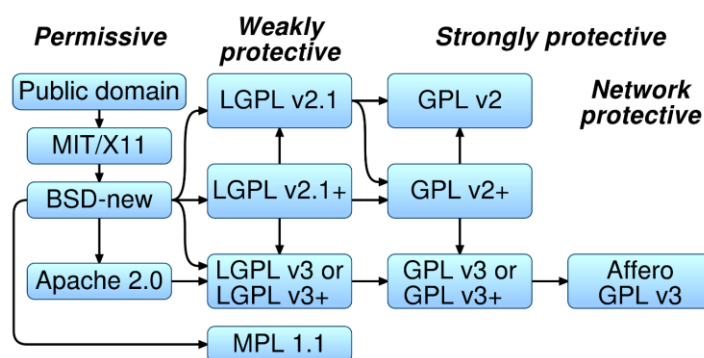


Figure 13 Floss-license-compatibility (Source: [Wikimedia Commons](https://commons.wikimedia.org/wiki/File:Floss-license-slide-image.svg)⁷³)

⁷² https://en.wikipedia.org/wiki/Permissive_software_license

⁷³ <https://commons.wikimedia.org/wiki/File:Floss-license-slide-image.svg>

As far as the technical outputs depend themselves for other components that might be (or might be not) more or less permissive, we need take note of backwards compatibility.

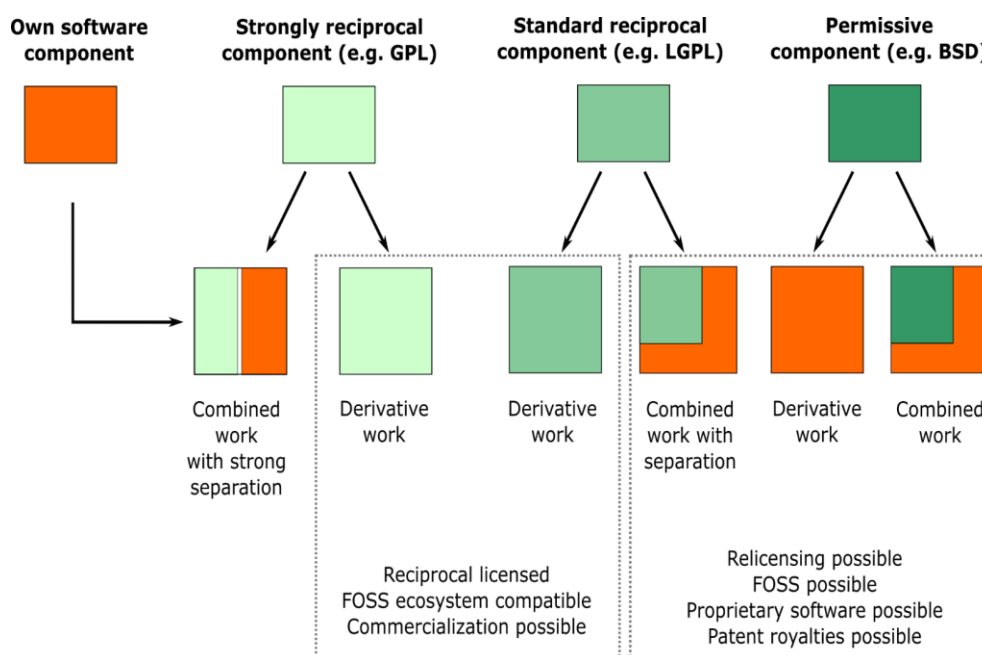


Figure 14 Software license compatibility for derived works (Source: [Wikimedia Commons](#)⁷⁴)

5.3 Annex III: Resources

Project sites

- [PoliRural](#) - Future Oriented Collaborative Policy Development for Rural Areas and People
- [Digital Innovation Hub](#) – Social Space for Smart Regions
- [PoliRural GitHub Organization](#) - WP3 - Final Integration, Release & Licensing

Main sites related to this document

- [GitHub](#) / [GitHub Docs](#)
- [OpenAIRE](#)
- [OpenAIRE in EOSC](#)
- [Zenodo](#)

Papers, reports and informative sites

- [Research Data Guide 2022](#) – EUI Library
- [Open Source and Open Data Licenses in the Smart Infrastructure Era: Review and License Selection Frameworks](#) | SpringerLink – Pay walled
- [Metadata quality](#) | Nature collections

⁷⁴ <https://commons.wikimedia.org/wiki/File:Software-license-compatibility-graph.svg>

- [FAIRsharing.org](https://fairsharing.org) – *Standards, databases and policies*
- [Open Source Licensing primer for Enterprise AI/ML | Towards Data Science](#)
- [How to Apply a License to Your Open Source Software Project | FOSSA](#)
- [Disciplinary repositories](#) - *Open Access Directory at Simmons University*
- [License compatibility | Wikipedia](#)
- [Permissive software license | Wikipedia](#)

Tools

- [Choose an open source license | Choose a License](#) – *by GitHub*
- [JSON-LD Playground](#)
- [JSON Formatter & Validator \(curiousconcept.com\)](#)
- [Citation File Format \(CFF\) \(citation-file-format.github.io\)](#)
- [ScienceOpen](#) – *An interactive discovery environment*