

Archive your software to Software Heritage.

The example of an EOSC-Core service

Themis Zamani (GRNET)

Agelos Tsalapatis (GRNET)

Kostas Koumantaros (GRNET)



Funded by
the European Union

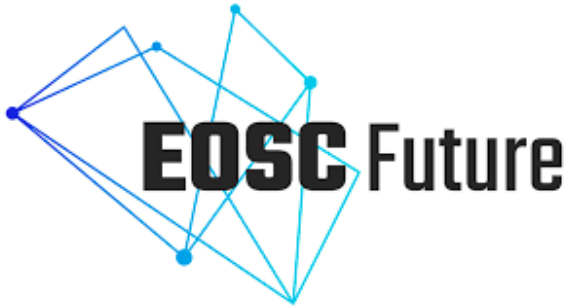
WP6 Partners





Set up of **regular archival** in Software Heritage for **EOSC-core** code hosting platforms

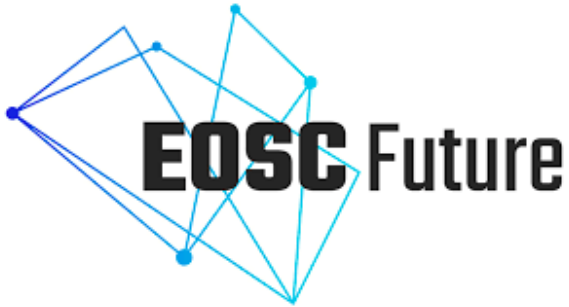
Objectives



Set up of **regular archival** in
Software Heritage for **EOSC-core**
code hosting platforms

**Find
repositories**

Objectives



Set up of **regular archival** in
Software Heritage for **EOSC-core**
code hosting platforms

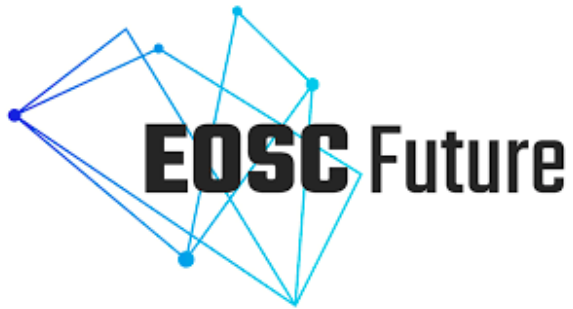


Software Heritage

**Find
repositories**

How to archive

Objectives

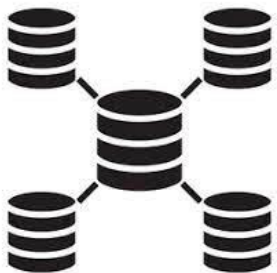


Set up of **regular archival** in
Software Heritage for **EOSC-core**
code hosting platforms



Software Heritage

repositories



archival



archive





The Archival Script

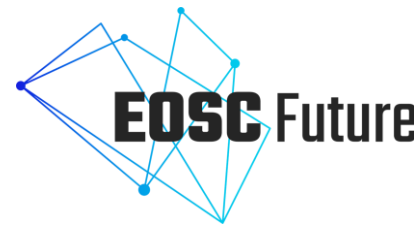
The What

Archiving EOSC Core components repositories

The script reads a list of repositories sources alongside with their type (git,svn.etc) and attempts to archive each single one of them through the software heritage api.



Prerequisites : The List



Prerequisites : Understand the common functions





The Archival Script

Archiving EOSC Core components repositories

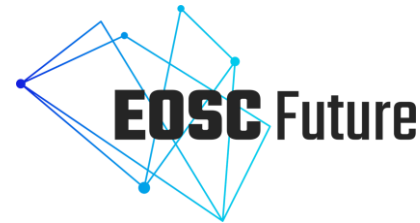
The Process

- Get a list of repositories to check
- Use the visit api call to determine if the repository is already archived.
- Use the save api call to either get the status or begin the process of archival for a missing one.
- Produce a report containing required information

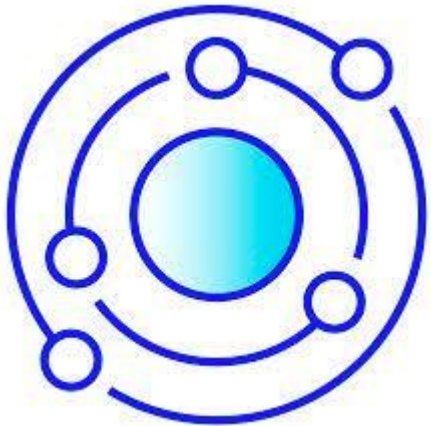


Next Steps

- Archive the repositories with the related metadata. (deposit action)
- Get correct list of repos



Main functionalities



The repository archiving script is a bridging component that provides

The following functionality:

- Reads as file input a list of version control urls that correspond to various repositories and will attempt to que them for archival through the software heritage API
- The first action is to utilise the **VISIT** api call that returns information about the archival of a repository.
- If a repository can be visited, we collect some additional information through the **STATUS** api call that will used in the last action of the script .
- In the other hand , if the repository does not have any information present in the software heritage system , we use the **ARCHIVE** api call to que the repository for archival .Repositories from known sources like GitHub are picked up and queued for archival automatically.
- The final step of the script is the compilation of a results file report that contains information about the status of the archival of all the provided repositories.



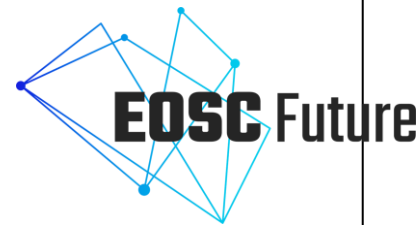
The Archival Report

The Report

The report is a file where its line represents the archival data of one of the provided repositories

1: **4066**, <https://example.com>, **git**, **accepted**, **not yet scheduled**, **2023-06-24T13:37:17**

- **The id of the archival request assigned by the SWH**
- **The url of the repo to be archived**
- **The version control system**
- **The status of the request**
- **The status of the archival**
- **The timestamp of the archival request**



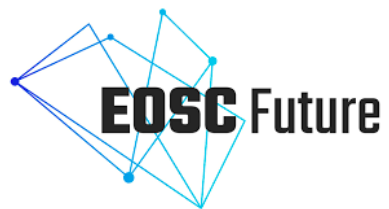


1st question

how can we search for the new repositories?

2nd question

how can we add more information for the repositories so that everyone can easily find the code repository based on various metadata?



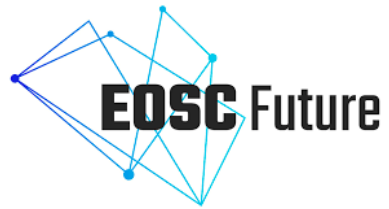
Software Heritage

<https://archive.softwareheritage.org/browse/search/>

1st question how can we search for the new repositories?

The screenshot shows the Software Heritage search interface. The search term 'themis' is entered in the search bar. A dashed box highlights the search filters: 'only show origins visited at least once' (checked), 'filter out origins with no archived content' (checked), and 'search in metadata (instead of URL)' (unchecked). The search results table is as follows:

Origin type	Origin url	Archiving status
git	https://github.com/LFernandoC/themis	✓ Archived
git	https://github.com/reference-project/themis	✓ Archived
git	https://github.com/hgfeaon/themis	✓ Archived
git	https://github.com/Mohammadreza-M/themis	✓ Archived



2nd question

how can i add more information for the repositories so that everyone can easily find the code repository based on these description ?

LICENSE	-rw-r--r--	25.8 KB
Makefile	-rw-r--r--	439 bytes
README.md	-rw-r--r--	7.1 KB
codemeta.json	-rw-r--r--	1.5 KB



What is CodeMeta in Software Heritage?

Easy to discover the software projects you may be interested in

At Software Heritage, they decided early on to use the CodeMeta vocabulary internally when indexing metadata.

The adoption of the CodeMeta representation, in the form of codemeta.json files, for describing landmark legacy software in the framework of the Software Heritage Acquisition Process

I am a repository owner

Is the script responsible for everything? What should i do?

The answer

Create the metadata.json file for your repository



The Service Owners are the ones that know the information

It is a 10 minutes work to make your software famous :)

CodeMeta Generator <https://codemeta.github.io/codemeta-generator/>

CodeMeta generator

Most fields are optional. Mandatory fields will be highlighted when generating Codemeta.

<p>The software itself</p> <p>Name <input type="text" value="My Software"/> <small>the software title</small></p> <p>Description <input style="width: 100%; height: 40px;" type="text" value="My Software computes ephemerides and orbit propagation. It has been developed from early '80."/></p> <p>Creation date <input type="text" value="YYYY-MM-DD"/></p> <p>First release date <input type="text" value="YYYY-MM-DD"/></p> <p>License(s) <input type="text"/> <small>from SPDX licence list</small></p>	<p>Discoverability and citation</p> <p>Unique identifier <input type="text" value="10.151.xxxxx"/> <small>such as ISBNs, GTIN codes, UUIDs etc.. http://schema.org/identifier</small></p> <p>Application category <input type="text" value="Astronomy"/></p> <p>Keywords <input type="text" value="ephemerides, orbit, astronomy"/></p> <p>Funding <input type="text" value="PRA_2018_73"/> <small>grant funding software development</small></p> <p>Funder <input type="text" value="Università di Pisa"/> <small>organization funding software development</small></p> <p>Authors and contributors can be added below</p>	<p>Development community / tools</p> <p>Code repository <input type="text" value="git+https://github.com/You/RepoName.git"/></p> <p>Continuous integration <input type="text" value="https://travis-ci.org/You/RepoName"/></p> <p>Issue tracker <input type="text" value="https://github.com/You/RepoName/issues"/></p> <p>Related links <input style="width: 100%; height: 40px;" type="text"/></p>
<p>Run-time environment</p> <p>Programming Language <input type="text" value="C#, Java, Python 3"/></p> <p>Runtime Platform</p>	<p>Current version of the software</p> <p>Version number <input type="text" value="1.0.0"/></p> <p>Release date</p>	<p>Additional Info</p> <p>Reference Publication <input type="text" value="https://doi.org/10.1000/xyz123"/></p> <p>Development Status</p>

CodeMeta metadata

The Software itself



Syntax:

```
https://api.github.com/repos/{:owner}/{:repository}
```

example: <https://api.github.com/repos/argoeu/argo-messaging>



```
svn info filename
```

```
svn propget --revprop -r 0 svn:date
```

```
http://svn.apache.org/repos/asf
```

```
"html_url": "https://github.com/ARGOeu/argo-messaging",  
"description": "The ARGO Messaging API is implemented as  
a Publish/Subscribe Service. Instead of focusing on a  
single Messaging API specification for handling the logic  
of publishing/subscribing to the broker network the API  
focuses on creating nodes of Publishers and Subscribers  
as a Service.",
```

The software itself

full_name

Name

the software title

Description

Creation date

First release date

License(s)

from [SPDX licence list](#)

Use [SPDX license identifier](#)

CodeMeta metadata / GITHUB names

Property	GitHub
codeRepository	html_url
programmingLanguage	languages_url
downloadUrl	archive_url
author	login
dateCreated	created_at
dateModified	updated_at
license	license
description	description
identifier	id
name	full_name
issueTracker	issues_url

CodeMeta metadata

The Authors / Contributors

You can add as many

- authors
- contributors

as you want.

Authors

Contributors

contributors

Order of contributors does not matter.

Given name

Family name

E-mail address

URI

Affiliation

CodeMeta metadata

Discoverability /. Citation

This is information that the service owner should add so that potential users / developers can see

- application category
- keywords
- Funding
- Funder

Discoverability and citation

Unique identifier ^{id}

10.151.xxxxx

such as ISBNs, GTIN codes, UUIDs etc.. <http://schema.org/identifier>

Application category

Astronomy

Keywords

ephemerides, orbit, astronomy

Funding

PRA_2018_73

grant funding software development

Funder

Università di Pisa

organization funding software development

CodeMeta metadata

Development community / tools

This information can be added

- from the repo automatically
- manually by the service owner

Development community / tools

Code repository **html_url**

Continuous integration

Issue tracker **issues_url**

Related links

CodeMeta metadata

Run-time Environment

This information can be added

- from the repo automatically
- manually by the service owner



Syntax:

<https://api.github.com/repos/{:owner}/{:repository}/languages>

```
{  
  "Go": 825574,  
  "Python": 43705,  
  "JavaScript": 7956,  
  "HTML": 3458,  
  "Makefile": 1634,  
  "CSS": 1596  
}
```

Run-time environment

Programming Language

C#, Java, Python 3

Runtime Platform

.NET, JVM

Operating System

Android 1.6, Linux, Windows, macOS

Other software requirements

Python 3.4
<https://github.com/psf/requests>

CodeMeta metadata

Current Version

This information is used to have a view of the current version of the software .

Current version of the software

Version number `html_url`

1.0.0

Release date

YYYY-MM-DD

Download URL

<https://example.org/MySoftware.tar.gz>

Release notes

```
Change log: this and that;  
Bugfixes: that and this.
```

CodeMeta metadata

Additional Info

This information is used to connect sub-components of a specific component , or to connect to a bigger project

Status

- **WIP** – Initial development is in progress, but there has not yet been a stable, usable release suitable for the public.
- **Active** – The project has reached a stable, usable state and is being actively developed.
- **Inactive** – The project has reached a stable, usable state but is no longer being actively developed; support/maintenance will be provided as time allows.

Additional Info

Reference Publication

<https://doi.org/10.1000/xyz123>

Development Status

see www.repostatus.org for details

Is part of

<http://The.Bigger.Framework.org>

CodeMeta JSON File

The EOSC Messaging example

CodeMeta generator

Most fields are optional. Mandatory fields will be highlighted when generating CodeMeta.

<p>The software itself</p> <p>Name Argo Messaging Service</p> <p>Description The Argo Messaging Service is a Publish/Subscribe service, which implements the Google Pubsub protocol. Instead of focusing on a single Messaging API specification for handling the logic of publishing/subscribing to the broker network the API focuses on/</p> <p>Creation date 2015-12-20</p> <p>First release date 2016-03-24</p> <p>License(s) from SPDX license list Apache-2.0.html Remove</p>	<p>Discoverability and citation</p> <p>Unique identifier 10.151.xxxxxx such as ISBNs, GTIN codes, UUIDs etc. http://uberon.org/identifier</p> <p>Application category messaging service</p> <p>Keywords messaging, pub/sub, golang, kafka</p> <p>Funding PIRA_2018_73 grant funding software development</p> <p>Funder GRNET S.A. - National Infrastructures for Research and Tec organization funding software development</p> <p>Authors and contributors can be added below</p>	<p>Development community / tools</p> <p>Code repository https://github.com/ARGOeu/argo-messaging.git</p> <p>Continuous integration https://travis-ci.org/YouRepoName</p> <p>Issue tracker https://github.com/ARGOeu/argo-messaging/issues</p> <p>Related links</p>	<p>Run-time environment</p> <p>Programming Language Golang, Python, Java</p> <p>Runtime Platform NET, JVM</p> <p>Operating System Android 1.6, Linux, Windows, macOS</p> <p>Other software requirements Python 3.4 https://github.com/paf/request</p>	<p>Current version of the software</p> <p>Version number 1.5.0</p> <p>Release date 2023-05-23</p> <p>Download URL https://github.com/ARGOeu/argo-messaging/archive/refs/heads/master.zip</p> <p>Release notes Update README.md Create CITATION.cff Create CONTRIBUTING.md Create CODE_OF_CONDUCT.md</p>	<p>Additional Info</p> <p>Reference Publication https://doi.org/10.1000/kyz123</p> <p>Development Status active www.repostatus.org for details</p> <p>Is part of http://The.Bigger.Framework.org</p>
---	--	--	---	---	--

Authors

Add one | Remove last

<p>Author #1</p> <p>Given name Themis</p> <p>Family name Zamani</p> <p>E-mail address themis@admin.grnet.gr</p> <p>URI https://orcid.org/0000-0003-1148-1736</p> <p>Affiliation GRNET S.A.</p>	<p>Author #2</p> <p>Given name Konstantinos</p> <p>Family name Kagkaidis</p> <p>E-mail address kagka@admin.grnet.gr</p> <p>URI http://orcid.org/0000-0002-1825-0097</p> <p>Affiliation GRNET S.A.</p>	<p>Author #3</p> <p>Given name Agielos</p> <p>Family name Tsapatidis</p> <p>E-mail address agelostsa@admin.grnet.gr</p> <p>URI http://orcid.org/0000-0002-1825-0097</p> <p>Affiliation GRNET S.A.</p>	<p>Author #4</p> <p>Given name Kkoumas</p> <p>Family name Koumantaros</p> <p>E-mail address kkoum@admin.grnet.gr</p> <p>URI http://orcid.org/0000-0002-1825-0097</p> <p>Affiliation GRNET S.A.</p>
--	--	--	---

Contributors

Order of contributors does not matter.

Add one | Remove last

<p>Contributor #1</p> <p>Given name Konstantinos</p> <p>Family name Evangelou</p> <p>E-mail address kevangel@admin.grnet.gr</p> <p>URI http://orcid.org/0000-0002-1825-0097</p> <p>Affiliation GRNET S.A.</p>
--

CodeMeta JSON File The EOSC Messaging example

```
"@context": "https://doi.org/10.5063/schema/codemeta-2.0",  
"@type": "SoftwareSourceCode",  
"license": "https://spdx.org/licenses/Apache-2.0.html",  
"codeRepository": "https://github.com/ARGOeu/argo-messaging.git",  
"dateCreated": "2015-12-20",  
"datePublished": "2016-03-24",  
"dateModified": "2023-05-23",  
"downloadUrl": "https://github.com/ARGOeu/argo-messaging/archive/refs/heads/master.zip",  
"issueTracker": "https://github.com/ARGOeu/argo-messaging/issues",  
"name": "Argo Messaging Service",  
"version": "1.5.0",
```

"description": "The ARGO Messaging Service is a Publish/Subscribe Service, which implements the Google PubSub protocol. Instead of focusing on a single Messaging API specification for handling the logic of publishing/subscribing to the broker network the API focuses on creating nodes of Publishers and Subscribers as a Service. It provides an HTTP API that enables Users/Systems to implement message oriented services using the Publish/Subscribe Model over plain HTTP. In the Publish/Subscribe paradigm, Publishers are users/systems that can send messages to named-channels called Topics. Subscribers are users/systems that create Subscriptions to specific topics and receive messages."

```
"applicationCategory": "messaging service",  
"developmentStatus": "active",
```

```
"isPartOf": "https://eosc-portal.eu",
```

```
"funder": {  
  "@type": "Organization",  
  "name": "GRNET S.A. – National Infrastructures for Research and Technology"  
}
```

CodeMeta JSON File

The EOSC Messaging example

```
"keywords": [
  "messaging", "pub/sub", "golang", "kafka"
],
"programmingLanguage": [
  "Golang", "Python", "Java"
],
"author": [
  {
    "@type": "Person",
    "@id": "https://orcid.org/0000-0003-1148-1738",
    "givenName": "Themis",
    "familyName": "Zamani",
    "email": "themis@admin.grnet.gr",
    "affiliation": { "@type": "Organization", "name":
"GRNET S.A"
    }
  }, {
    "@type": "Person",
    "givenName": "Agelos",
    "familyName": "Tsalapatis",
    "email": "agelostsal@admin.grnet.gr",
    "affiliation": { "@type": "Organization", "name":
"GRNET S.A " }
  }
]
```

```
"@type": "Person",
"givenName": "Konstantinos",
"familyName": "Kagkelidis",
"email": "kaggis@admin.grnet.gr",
"affiliation": {
  "@type": "Organization",
  "name": "GRNET S.A "
}
},
],
"contributor": [
  {
    "@type": "Person",
    "givenName": "Konstantinos",
    "familyName": "Evangelou",
    "email": "kevangel@admin.grnet.gr",
    "affiliation": {
      "@type": "Organization",
      "name": "GRNET S.A"
    }
  }
]
```

What are the least data I should use

The codemeta file

All info is necessary

Create the metadata.json at least with:

The Software name

The Authors / Contributors

Discoverability / Citation

Additional Info



FAIRCORE4EOSC

Core Components Supporting a FAIR EOSC



faircore4eosc.eu



[@FAIRCORE4EOSC](https://twitter.com/FAIRCORE4EOSC)



[company/faircore4eosc](https://www.linkedin.com/company/faircore4eosc)



[FAIRCORE4EOSC](https://www.youtube.com/FAIRCORE4EOSC)

