The JERICO-S3 project is funded by the European Commission's H2020 Framework Programme under grant agreement No. 871153. Project coordinator: Ifremer, France.

D6.9 E-library contribution report

Joint European Research Infrastructure for Coastal Observatories
Science, Services, Sustainability

DELIVERABLE #, WP# and full title	JERICO-S3 D6.9 - WP6 - "E-library contribution report"
5 Key words	Data lifecycle, data management, metadata model, collection method
Lead beneficiary	MARIS
Lead Author	Peter Thijsse (MARIS)
Co-authors	Miguel Charcos Llorenz (SOCIB)
Contributors	
Final version date/ Submission date (dd.mm.yyyy)	09.04.2024

Nature: **R**

(R = Report, P = Prototype, D = Demonstrator, O = Other)

Dissemination level : **PU**

PU = Public, PP = Restricted to other programme participants (including the Commission Services), RE = Restricted to a group specified by the consortium (including the Commission Services), CO = Confidential, only for members of the consortium (including the Commission Services)

GRANT N°: 871153
PROJECT ACRONYM : JERICO-S3
PROJECT NAME : Joint European Research Infrastructure for Coastal Observatories - Science, services, sustainability
COORDINATOR : Laurent DELAUNEY - Ifremer, France - jerico@ifremer.fr

DOCUMENT TECHNICAL DESCRIPTION

Document ID	JERICO-S3-WP6-D6.11-09.04.2024-V0.9

REVISION HISTORY

Revision	Date	Modification	Author
V0.1	13/02/2024	First draft	Peter Thijsse
V0.2	01/03/2024	Updated version	Miguel Charcos Llorenz
V0.3	18/03/2024	Review and edits	Peter Thijsse
V0.4	20/03/2024	Updated version	Miguel Charcos Llorenz
V0.9	09/04/2024	Final draft, ready for review	Peter Thijsse

APPROVALS

	Name	Organisation	Date	Visa
Coordinator	Delauney Laurent	Ifremer	09/04/2024	X
WP Leaders	Peter Thijsse	MARIS	09/04/2024	X

Diffusion list

Consortium beneficiaries	Third parties	Associated Partners	other
X	X		

TABLE OF CONTENT

List of frequently used acronyms

BC2026: Blue-Cloud 2026 project
CMEMS: Copernicus Marine Environment Monitoring Service
DMP: Data Management Plan
DOI: Digital Object Identifier
EMODNet: European Marine Observation and Data Network
EOSC: European Open Science Cloud
FAIR/FAIRness: Findable Accessible Interoperable Reusable (data, software, service)
IRS: Integrated Regional Site
JERICO-CORE: JERICO Coastal Ocean Resource Environment
JERICO-RI: Jerico Research Infrastructure
NVS: NERC Vocabulary Service - important vocabulary with ontology underpinning metadata in SeaDataNet, EMODnet and other.
PSS: Pilot Super Site
RDM: Research data management
QC: Quality Control
SDN: SeaDataNet
SW: Software
VRE: Virtual Research Environment

1. EXECUTIVE SUMMARY

In the context of the JERICO-S3 project, WP6 aims to enhance the management and utilisation of coastal platforms and data. It addresses the need for streamlined data lifecycle management practices, from acquisition to publishing, stressing the commitment to data FAIRness and development of Best Practices. Within this framework, Task 6.4 is focusing on data management support activities. Task 6.4.1 is dedicated to the identification, adoption, and creation of Best Practices in data management, and one of the sub-tasks focuses on the compilation of a data management software catalogue.

This document provides an overview of the actions done to create a metadata model for a software description in several iterations. The actions started with the development of a first metadata model and the catalogue itself. A request followed to the JERICO community to supply insight in existing data management software using a spreadsheet with the metadata format in it. This provides a useful insight which is shared in chapter 3. Using this input a second iteration has been done, with an updated metadata model, and a more sustainable method to harvest the descriptions from data providers. This work in JERICO-S3 is a response to the essential role of software solutions in facilitating efficient data management practices across selected platform and sensor types. By exploring, documenting, compiling and sharing existing software solutions—ranging from Jupyter notebook scripts for Quality Control (QC) to data harmonisation scripts— this task has created a robust foundation for a repository of tools, to be shared, discussed and used in the JERICO-RI coastal data community.

In the document we have described the approach to develop the initial software e-library and its first content (chapter 3), an improved iterated version of the metadata model and collection method (chapter 4), an analysis of the automatically collected content (chapter 5), followed by conclusions and a forward look in chapter 6.

2. INTRODUCTION

In the context of the JERICO-S3 project, WP6 aims to enhance the management and utilisation of coastal platforms and data. It addresses the need for streamlined data lifecycle management practices, from acquisition to publishing, emphasising the commitment to data FAIRness and development of Best Practices. Within this framework, Task 6.4 is focusing on data management support activities. These activities are designed not only to complement the efforts under Tasks 6.2 and 6.3, but also to facilitate a comprehensive approach to data management across the JERICO-RI coastal platforms and stimulate sharing of experiences in the RI.

Task 6.4.1 is dedicated to the identification, adoption, and creation of Best Practices in data management, aligning closely with the needs of Pilot Super Sites (PSS) and Integrated Regional Systems (IRSs). For this purpose, task 6.4.1.2 specifically addresses the compilation of data management software. This initiative recognizes the essential role of software solutions in facilitating efficient data management practices across selected platform and sensor types. By exploring, documenting, and compiling existing software solutions—ranging from Jupyter notebook scripts for Quality Control (QC) to data harmonisation scripts—this task aims to create a robust repository of tools, to be shared, discussed and used in the community.

The JERICO Coastal Ocean Resource Environment (JERICO-CORE), the pilot of the e-JERICO infrastructure developed under JERICO-S3, stands as a pivotal central hub within the JERICO framework. The JERICO-CORE is designed to streamline the discovery, access, management, and interaction with an extensive array of JERICO resources. This comprehensive suite includes services, datasets, **software**, best practices, manuals, publications, organisations, projects, observatories, equipment, data servers, e-libraries, alongside support and training materials. The inception of a knowledge base catalogue to aggregate these resources, while not a novel idea, embodies a significant leap forward in enhancing the accessibility and utility of JERICO's vast assets. In this context, it is an important tool to support the activities of task 6.4.1, and activities in 6.4.1 support the content.

In the next sections we will describe the approach to develop the software e-library (chapter 3), a description and analysis of the content (chapter 4 and chapter 5) and conclusions and forward look (chapter 6).

3. E-Library approach and first iteration

3.1. E-Library as part of the JERICO-CORE

The development of the JERICO-CORE software catalogue provides a step forward in the aggregation and accessibility of software tools among JERICO partners. This catalogue is designed to capture various types of entities and facilitate their discovery and usage. An initial evaluation conducted with the partners provided insights into the availability of data management related software that could be integrated into the catalogue and the required metadata for the schema. This preliminary assessment was instrumental in shaping the first metadata model for the catalogue and identifying the primary sources of information.

Table 1: Initial metadata model of the software catalogue

COLUMN NAME	DEFINITION
Institution	Name of the institution owning the code
Specific Software (SW) Contact (email)	Email address of the person that is in charge of maintaining and sharing the code
SW Name	Name of the software
SW Description	Description of the software (purpose, framework, ...)
Software Theme	Type of application of the software
Maturity level	The level of maturity is given by the extent in which the software is used, number of users, status of the documentation, status of testing,...
Platform/Instrument Type (only if specific for one or more types)	If the data handles or processes data of one or more specific platforms, this column contains a comma separated values with the names of the type of platforms (or instruments)
Programming Language	The programing languages that were used in the code (or comma separated values)
Current Repository	The type of the repository that is used to version control and share the code
Link to online repository	The link of the repository hosting the code if it is available and public
SW Input or interface spec (standard)	The inputs that the code uses (text files, configuration files, Netcdf files, DB, command line parameters,...)
SW Output (standard)	The inputs that the code outputs (text files, configuration files, Netcdf files, DB inputs, console messages,...)
Current Access	Availability of the code (public, restricted, private,...)
Preferred Method for sharing	If you were willing to share this code in the e-JERICO, this column contains the way that you would be willing to provide. The answers could be any of the common platforms such as github, docker hub,... but also your own in-house repositories. Also explain if you are willing to share it in

	a community repository or you prefer to keep yours in case you already use this option (e.g. github)
Comments	Write any comment concerning your software or notes about the information that you entered for this software. Feedback about the survey is also welcome in this column.

The metadata schema for software underwent significant evolution as our understanding of the partners' software landscape deepened, alongside the emerging needs for the assessment of software tools. In the initial phase of this exploration, we delineated specific fields in Table 1, marking our first step towards understanding the software tools available to our partners.

Initial software collection actions

The results of the initial inventory, detailed in Appendix A, includes 26 distinct tools and their dissemination methods—predominantly through GitHub, alongside Bitbucket and GitLab (see figure 1). Figure 2 offers a breakdown of the programming languages utilised, such as Javascript, Python, Octave/Matlab, Java, C, and R, reflecting the diverse technological landscape within the JERICO community. Furthermore, figure 3 provides an overview of the eight organisations, including GBIF, SOCIB, UPC, Hereon, FAMRI, CNR-ISMAR, VLIZ, and AZTI that are sharing and cataloguing their software tools.

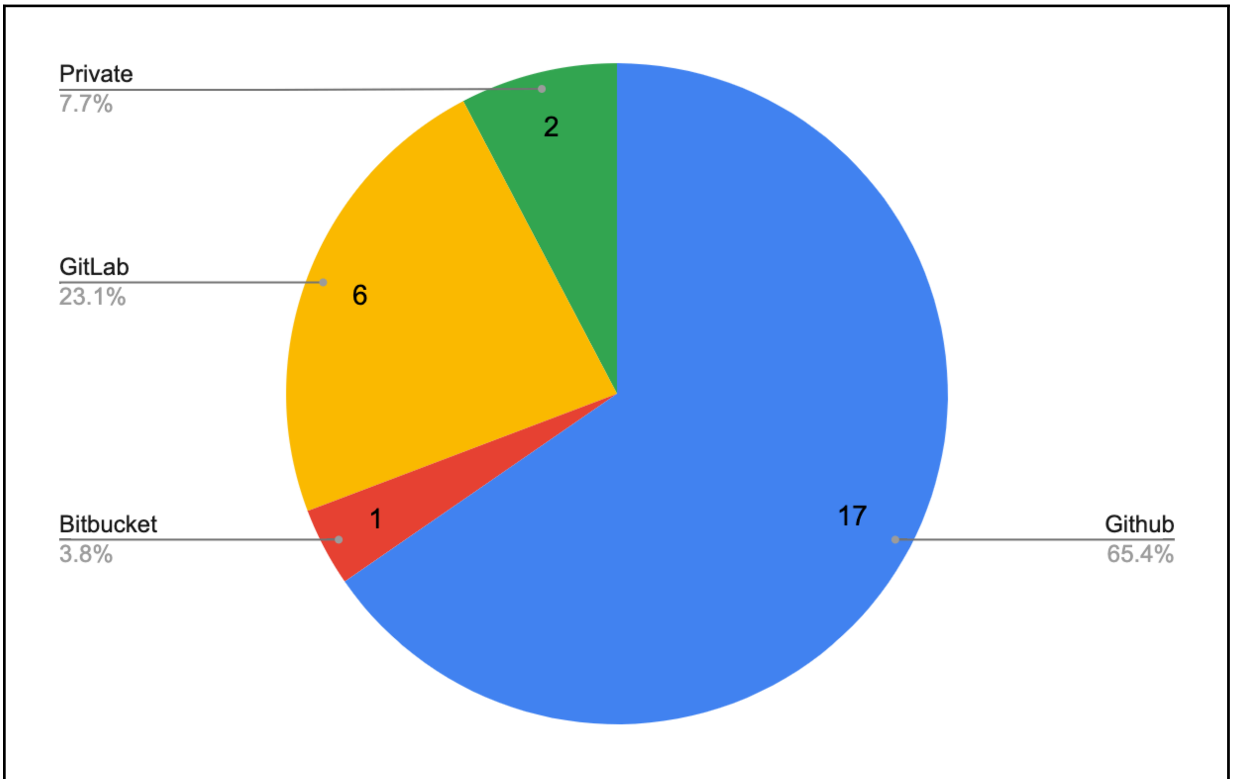

Figure 1: division of initial catalogue per repository

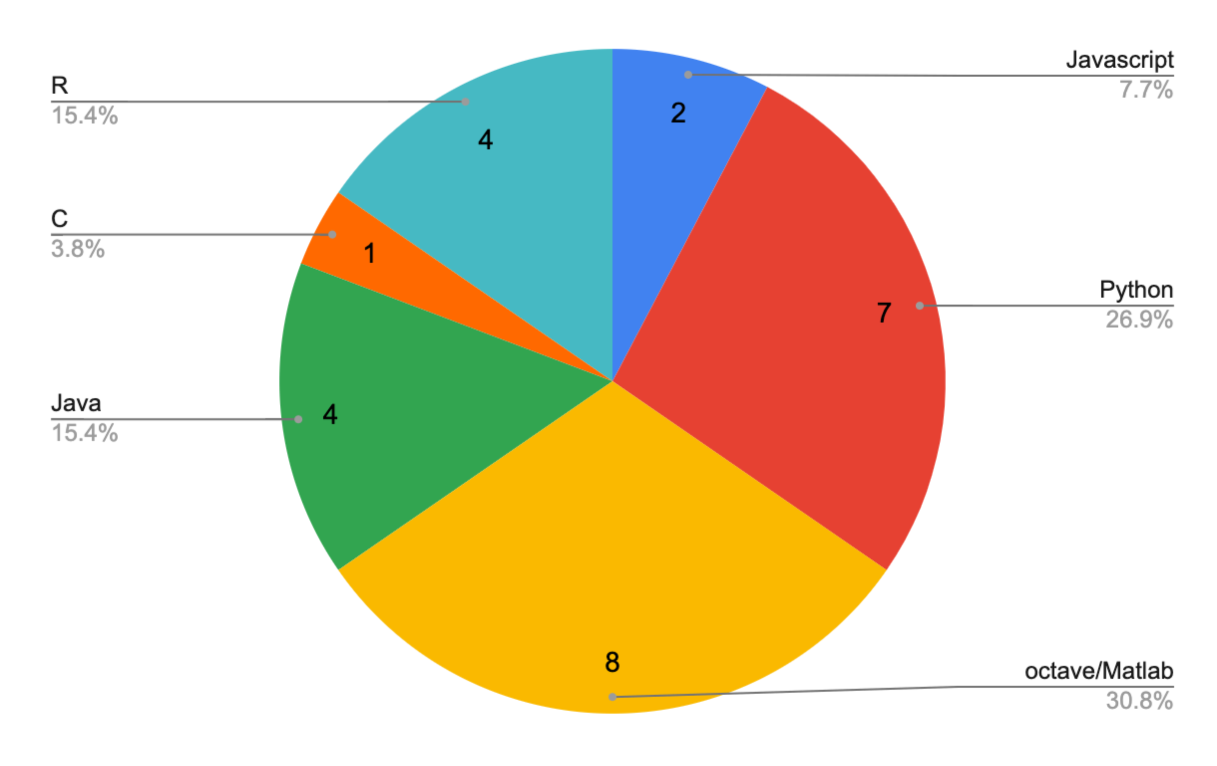

Figure 2: division of initial catalogue per coding language

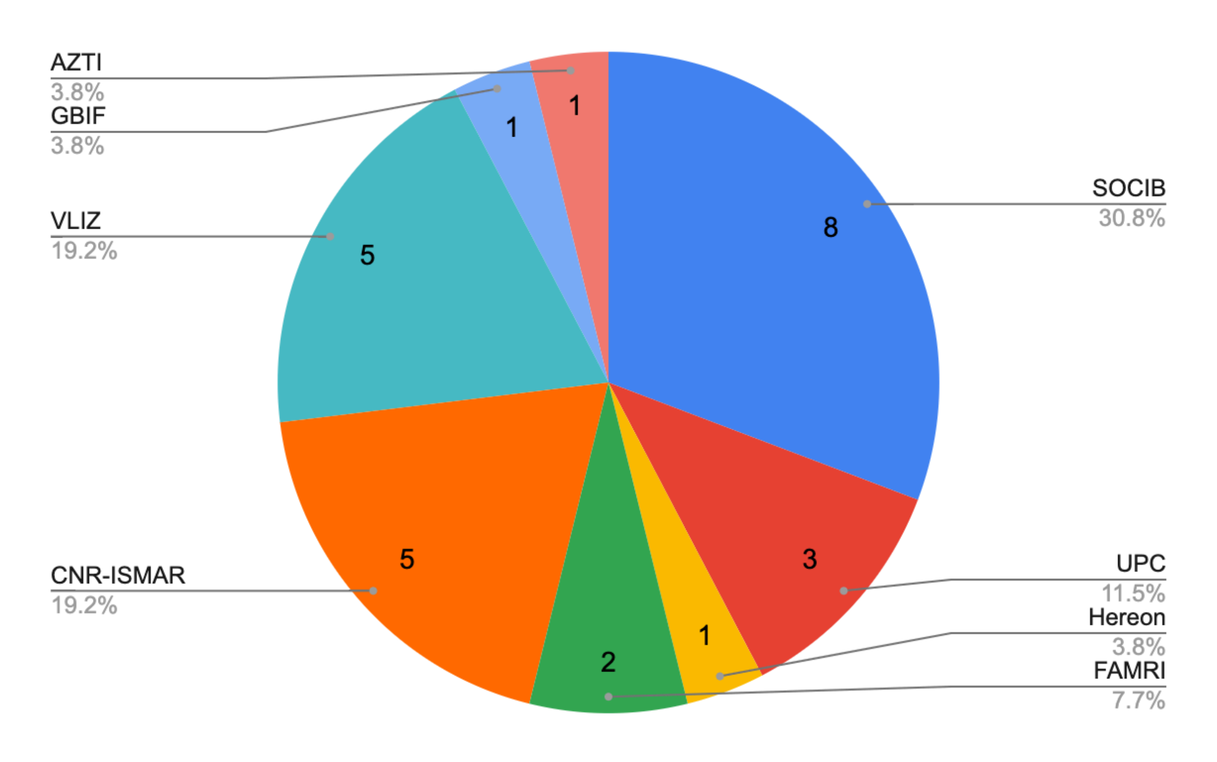

Figure 3: division of initial catalogue per provider

4. JERICO E-Library Catalogue - second iteration

As our collection efforts progressed, it became evident that a more robust metadata schema was necessary to accurately reflect the interconnection between software, e-libraries, data and other resources of the JERICO community. Detailed in Section 4, this enriched schema was the result of our collective experiences throughout the project. It extends beyond the initial metadata model and it was guided by the aggregation of software from repositories and in establishing links to resources from other information providers.

Moreover, we evolved the metadata schema further to incorporate details vital for broader contexts beyond the immediate scope of the JERICO-CORE pilot. While this advanced schema was not adopted within the pilot phase of JERICO-CORE, it is also outlined in this section, offering guidance as a strategic recommendation for forthcoming initiatives. This proposed schema serves not only as a guideline, but also as an indication for continuous improvement and adaptation. By promoting its adoption, we aim to facilitate a more informed, efficient, and cohesive approach to software integration and utilisation across the JERICO network, ensuring that future activities can benefit from a well-defined, comprehensive metadata framework. It is envisioned that this metadata framework will be improved in ongoing projects such as Blue Cloud 2026 and OSTrails that support interoperability with EOSC and other Blue Data Infrastructures.

4.1. E-Library collection process

In a strategic move to enhance the catalogue and ease/sustain the collection method, a decision was made to prioritise GitHub for collecting software. This process was executed in two phases: an initial call for software integration preceding the full operation of JERICO-CORE pilot under WP11, and a subsequent call in October 2023 aimed at enriching the catalogue with the latest tools before the culmination of the JERICO-S3 project. Partners were requested to tag their GitHub repositories with specific labels—either 'jerico-ri' or 'jerico-relevant'—to facilitate their identification and indexing within the catalogue. These labels served as markers, not indicating "created with JERICO-S3 funding", but of the tools' availability to the JERICO community. Additionally, partners were encouraged to incorporate a JSON file in their repositories to enrich the metadata with information that is not recorded in the github platform.

Taking into account the availability of valuable software beyond GitHub, a unique repository was established at https://github.com/socib/jerico-core-metadata to accommodate the manual inclusion of metadata through JSON files. The repository is equipped with templates to guide users in generating metadata for their resources, currently including both services and software definitions.

Deliverable D7.6 [1] outlines the overarching methodology of the resource information process for the JERICO-CORE pilot (see figure 4), where metadata from resource providers is accessed through their specific endpoints, for example using for software the GitHub API as detailed at https://docs.github.com/en/rest and documented above.

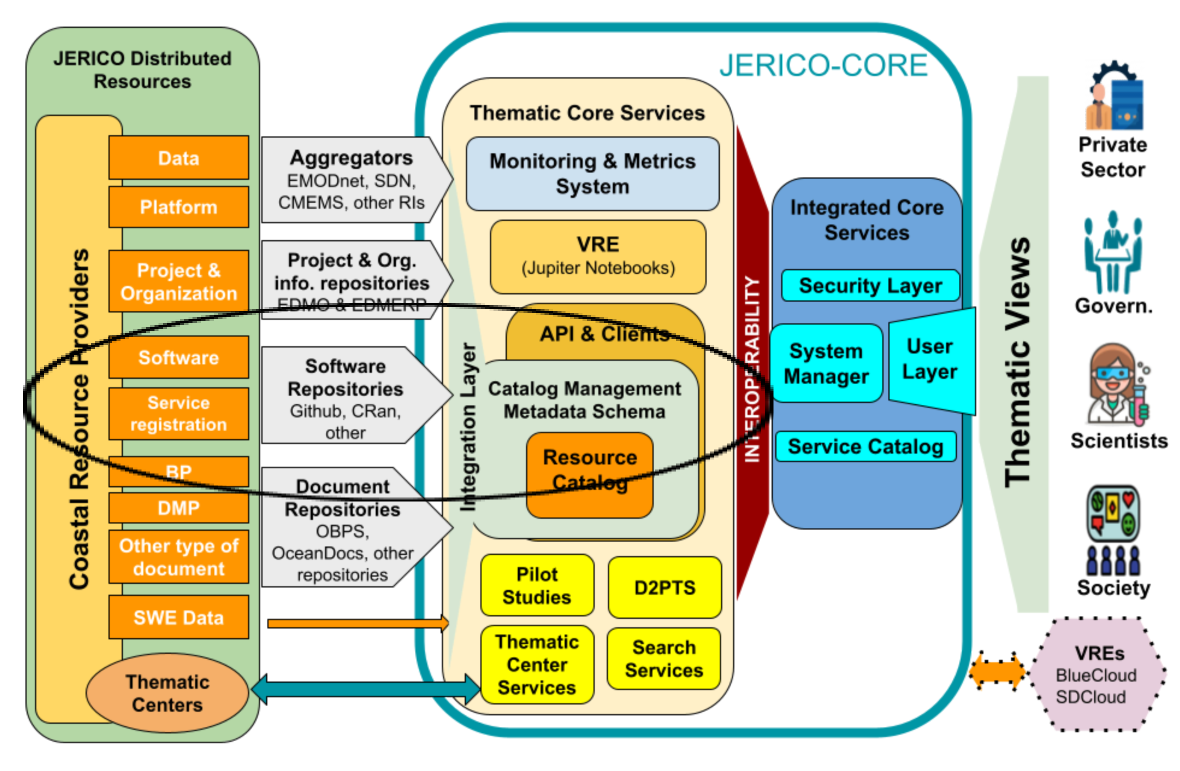

Figure 4: JERICO-CORE architecture - Software catalogues position

The methodology employed for information collection offers significant advantages. By integrating the harvesters of the JERICO infrastructure to operate automatically within the CORE (deployed e.g. partially in Blue-Cloud), JERICO partners gain the capability to index and update the metadata of tools in real-time.

This automation significantly reduces the manual effort previously required from partners to complete and update their information in an Excel table. Unlike this static and manual process, the dynamic approach of using labelled repositories and JSON files for metadata allows partners to initiate with a basic set of information and iteratively refine and expand the metadata over time. This flexibility means that initial metadata can be quickly created and then enhanced as more details become available or as the software evolves. Moreover, this method enables a more fluid and ongoing process of metadata management, where updates are directly and immediately reflected in the Virtual Research Environment, ensuring that the information is always current and reducing the time and effort involved in manual updates. While automatic metadata harvesting is not yet implemented in the JERICO-CORE pilot, this functionality is anticipated to be a cornerstone feature in the future e-JERICO infrastructure.

The development of the metadata schema (see table 2) was designed to integrate with the existing data providers, ensuring that adaptation of the source material was not necessary. However, the completion of the comprehensive metadata schema encountered obstacles, primarily due to the lack of a detailed definition of various components such as data flow and equipment type. The intricacy of accurately representing data flow for the purpose of automatic assessment and management proved to be a complex challenge that could not be fully addressed within the scope of the project. As a result, certain aspects of the schema remain underdeveloped, highlighting gaps that necessitate further exploration and refinement. This ongoing work is slated for continuation within the context of the BC2026 and OSTRails projects, where a deeper understanding of these components will be pursued. In the interim, the use of keywords has been employed to bridge the information gap, serving as a temporary measure to enhance the metadata's completeness and utility. This approach, while effective in the short term, underscores the need for a more thorough and nuanced development of the metadata schema to fully meet JERICO's needs.

Attribute	Description
abstract	A brief summary of the software, highlighting its purpose and core functionality.
creator	The person or organization that created the software.
concept	The main idea or the problem the software aims to address or solve.
description	A detailed description of the software, including its features, use cases, and benefits.
downloadURL	A URL where the software can be downloaded.
keywords	Keywords associated with the software to facilitate search and discovery.
language	The natural language(s) in which the software documentation or interface is available.
license	The license under which the software is released, specifying usage rights and restrictions.
memory_requirements	The minimum memory requirements needed to run the software.
name	The name of the software.
published	The date when the software was first made available.
operating_system	The operating system(s) required to run the software.
storage_requirements	The storage space required for installation and operation of the software.
processor_requirements	The type or specifications of the processor required to run the software efficiently.
programming_language	The programming language(s) in which the software is written.
runtime	The runtime environment required for the software to operate.
size	The file size of the software package.
suite	The software suite to which the software belongs, if applicable.
updated	The date when the software was last updated.
url	The official website or landing page for the software.
program	The overarching program or initiative the software is part of.
brand	The brand or product line the software is associated with.
spatial	The geographical latitude and longitude (mean, min and max) related to the software's use or application.
temporal	The start and end dates for the period during which the software is relevant or was used.
organization	The organization responsible for the software.
owner	The individual or organization that owns the software.
contributor	The individuals or organizations that contributed to the development of the software.
publisher	The individual or organization that published the software.
author	The primary authors (individual or organization) of the software.

distribution	Distribution of the software including access, download, mimetype.

Table 2: Iterated metadata model (first basis for the json file format)

4.2.Software metadata schema analysis

The analysis of the software metadata schema is structured around pivotal use cases that are essential for enhancing the accessibility, credibility, and utility of software within the JERICO community. This analysis takes into account the need for interoperability with European infrastructures and anticipates further exploration within the OSTrails project. OSTrails focuses on developing tools, services, and standards for Science Knowledge Graphs (SKG) and FAIRness measurements to improve interoperability across various research fields. These standards will support the implementation of machine actionable Data Management Plans (maDMPs) that facilitate the automatic management of data and measurement of impact along the data life cycle.

Therefore, these efforts align with the objective of Task 6.4 to support Best Practices in Data Management, making the analysis relevant to both current and future project phases. In the context of integrating software into automated workflows and enhancing data management practices, the proposed metadata schema is designed to support the seamless incorporation of software tools into larger systems or processes. This schema includes specific attributes aimed at facilitating the use of software within automated workflows, crucial for advancing research data management (RDM) and implementing effective Data Management Plans (DMPs) that embrace FAIR principles.

The analysis draws on key sources to ensure the proposed metadata schema will be interoperable across European infrastructures:

- **ENVRI Data Model**: As detailed in ENVRI-FAIR D7.3 (p21) [2], this model provides insights into data management practices that can inform the development of a robust metadata schema.
- **European Open Science Cloud (EOSC)**: Considerations for integration with EOSC standards and practices are crucial for ensuring broad accessibility and compliance.
- **Code Metadata Standards [3]**: Resources like the CodeMeta project offer valuable guidance on structuring metadata to promote discoverability and interoperability.

It leads to the following key aspects of this enhanced metadata schema to take into consideration:

- Discovering software
    - **Descriptive Elements**: Emphasise the software's title, keywords, and abstract to provide users with a clear understanding of its functionality and potential applications. They will also provide clear and concise **descriptions** of the software's functionality and potential use cases.

    - **Categorization**: Implement categories or domains within the metadata to help users easily locate software relevant to their research interests.

  - **Maturity Level**: Categorize software based on its development stage, from prototype to fully mature, guiding users in selecting tools that match their project's reliability and stability requirements.

- Credit for academic software
  - **Authorship**: Detailed **author** and **contributor** information prominently.
  - **Citation Guidelines**: Include instructions on how the software should be **acknowledged** or **cited** in publications, adding a field for the Digital Object Identifier (**DOI**) to facilitate this process.

- Replicating analysis
  - **Documentation**: Highlight the availability of user guides, manuals, and tutorials in the metadata. Encourage contributors to provide comprehensive **documentation to facilitate replication of analyses**.
  - **Dependencies**: Allow for the inclusion of detailed information on both internal and external **dependencies**.
  - **Integrity Measures**: Suggest providing **checksums** or **cryptographic signatures** with software downloads to ensure data integrity and security.

- Controlling versions and dependencies
  - **Version Information**: Ensure that **version** and **release** details are easily accessible within the metadata.
  - **Dependency Specification**: Clearly specify **dependencies**, including compatible versions, to aid in the management of software environments.
  - **Access to Versions**: Offer **links to download** specific software versions or reference version control repositories, such as Git, for access to **historical versions**.

- Integration into Automatic Processes
  - **Workflow Integration**: Metadata attributes that describe how software can be integrated into automated workflows, detailing the **steps**, **processes**, and **interfaces** required. This ensures clear **documentation** is available to facilitate the software's incorporation into broader research ecosystems.
  - **Equipment and Observing Platform Applicability**: Attributes specifying the **types of equipment** or observing platforms the software is designed for. This information is vital for researchers looking to apply software tools to specific datasets
  - **Data Lifecycle Stage Application**: Detailed metadata indicating the **stages** in the data cycle where the software can be applied, such as quality control (QC), format conversion, data processing, visualisation (viewer), data access, quality assessment reporting, and delayed mode analysis. This

clarity aids in selecting the appropriate tools for particular stages of the data lifecycle.

- **Support for FAIR Principles in DMPs**: Attributes that facilitate the evaluation of DMPs against FAIR principles to enhance the granularity, findability, accessibility, and preservation of DMPs. This involves metadata that can describe how software contributes to the creation, management, and assessment of DMPs. These aspects will be later defined in the OSTrails project and is tightly related to the description of JERICO resources in its resource catalogue and the European Open Science Cloud (EOSC).

This analysis results in the metadata schema proposed in Appendix B. By considering these use cases and sources, the schema will be positioned to support the effective management, sharing, and utilisation of software resources across diverse research contexts of JERICO. The latest is specially important in the current context of Digital Twin of the Ocean where integration of multidisciplinary data is essential to respond to users.

5. JERICO-CORE E-Library content analysis

We end the research part of the deliverable with analysis of the current content of the e-library.

5.1.E-Library Content

The catalogue was loaded from Github using the json metadata definition. The result of the catalogue included 35 e-libraries that were indexed at the time of writing. https://api.core.jerico-ri.eu/api/software/

When looking at the software providers, these were largely the same entities that participated in the initial survey.

- SOCIB
- UPC
- Hereon
- FAMRI
- CNR-ISMAR
- VLIZ
- GBIF
- AZTI

Per provider we received more contributions, but almost the same ratio was available again on Github (around 65%).

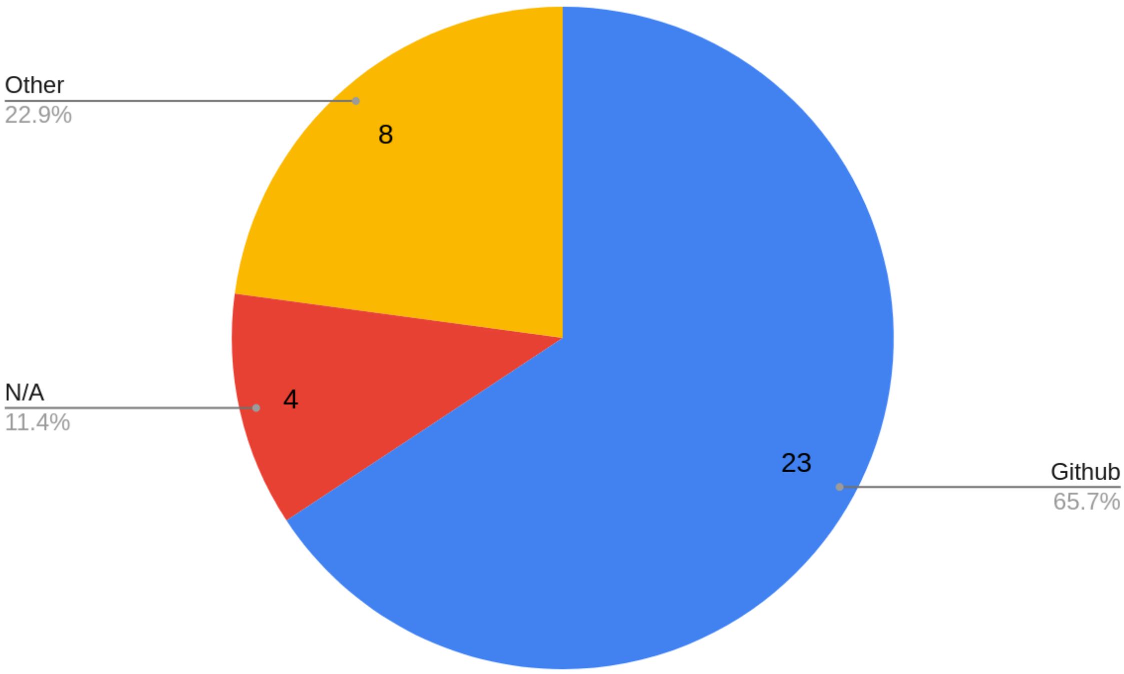

Figure 5: Ratio of contributions provided by code platforms

For the programming languages: Python and Matlab/Octave are the main contributions. Noticeable increase of C e-libraries.

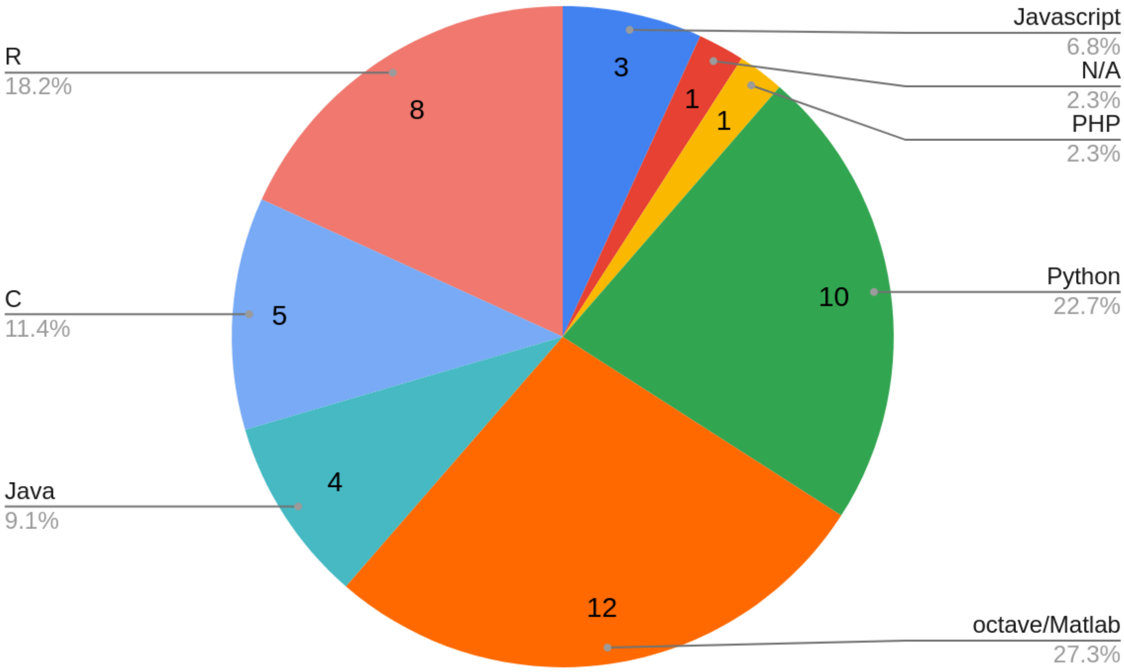

Figure 6: Division of the coding languages in the catalogue

An interesting viewpoint is to look at the contribution of the software to data management stages. Most of the contributions are related to format conversion, processing and data access. Only one is dedicated to delayed mode processing.

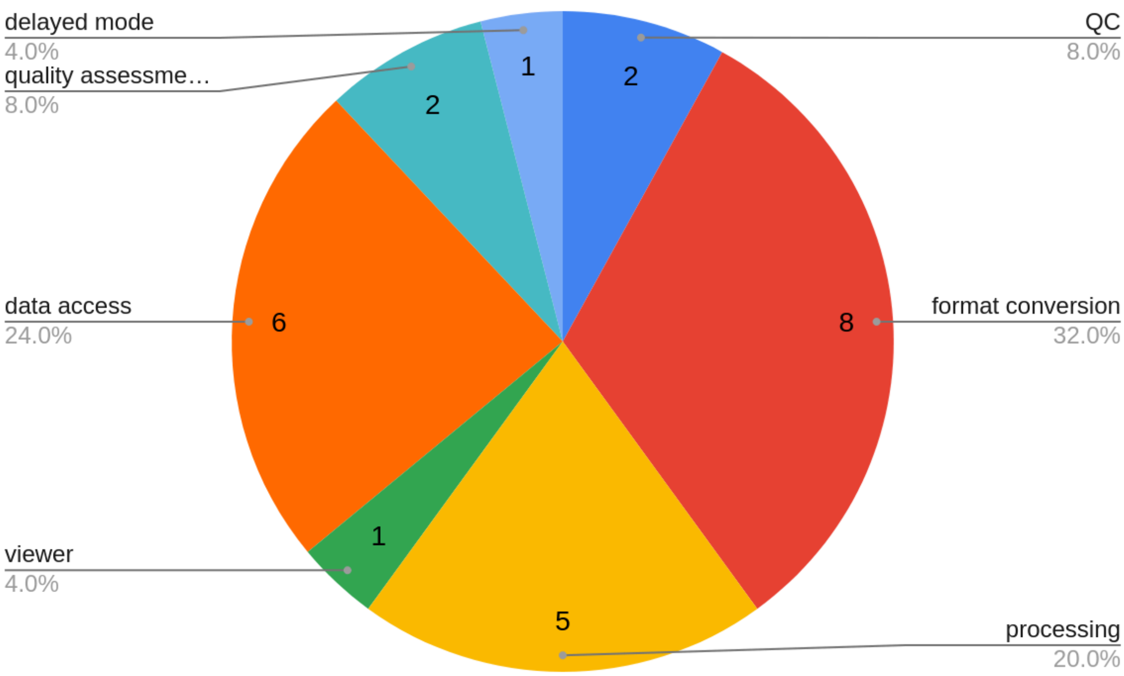

Figure 7: Division of the data management stages the software supports

If we look at the relevant equipment type the software supports, we can observe a bit of biassed division. HF Radar is the community that defined best their metadata, but also contributed most to this catalogue. Further we observe a good division over the equipment types, meaning that for all equipment software is really important.

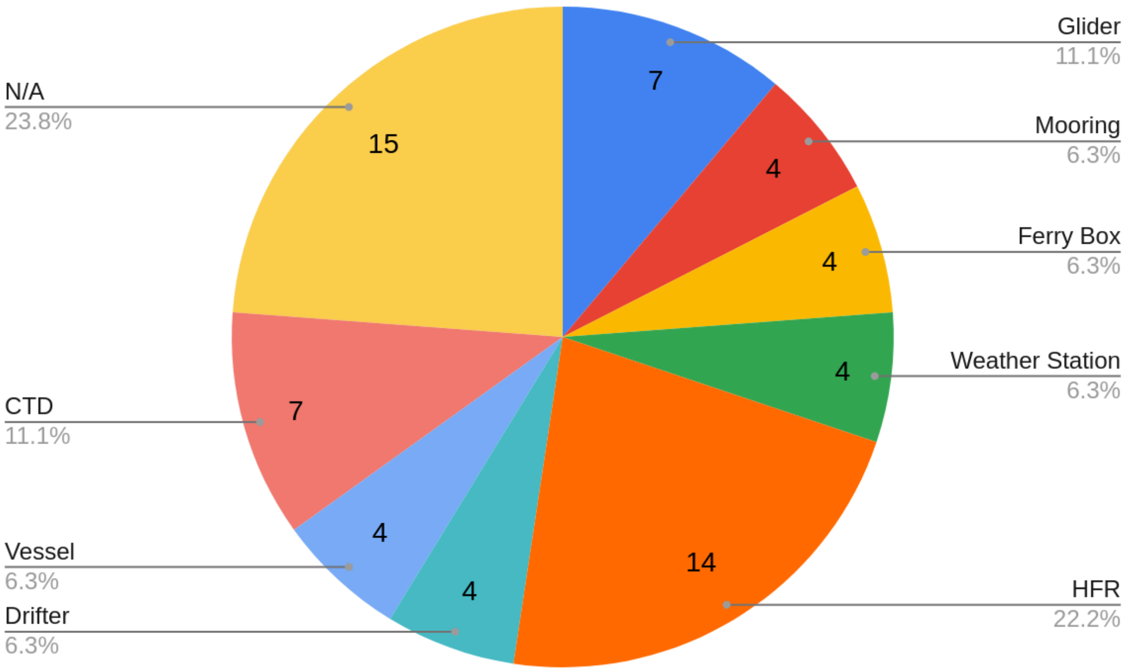

Figure 8: Division of the related equipment types

Very mature libraries are more than 55% available to the community, but there are also several single developer software solutions, ready to be shared..

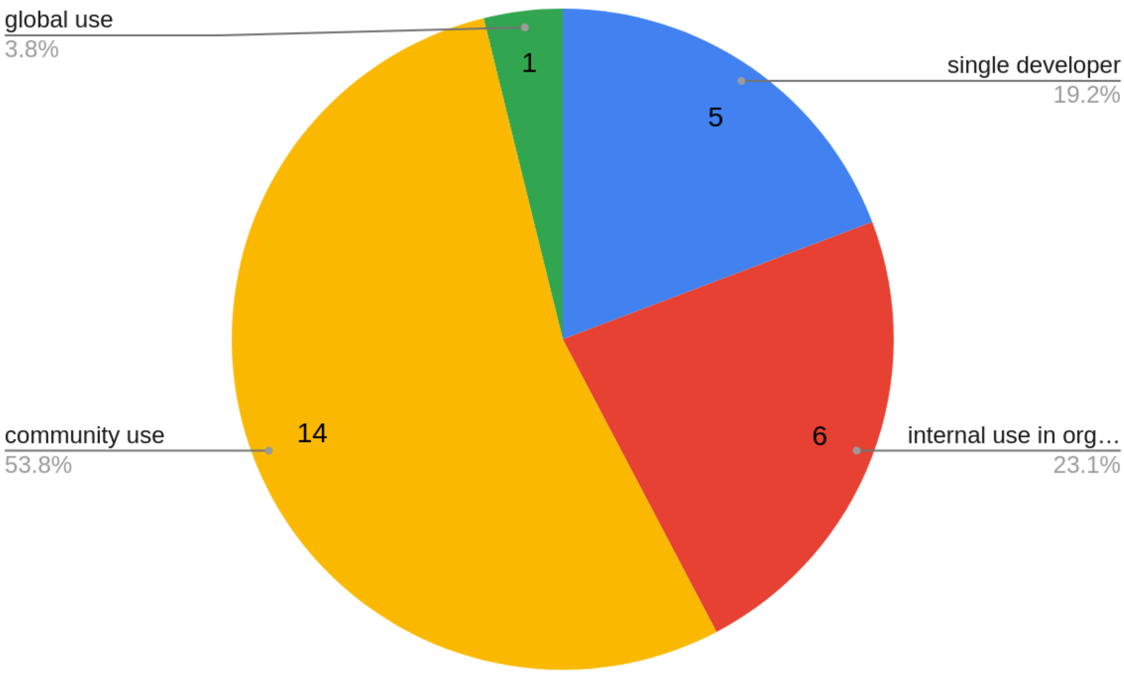

Figure 8: Division of the related equipment types

6. OUTREACH, DISSEMINATION AND COMMUNICATION ACTIVITIES

Further email requests for updates to the catalogue are foreseen ½ times per year.

The deliverable will be available to the full JERICO community, and published on Zenodo. It is aimed to trigger more interest in the subject and hopefully this will generate more interest to fill the content of the JERICO-CORE e-library, now and in future JERICO-RI developments.

7. CONCLUSIONS

This report has provided an overview of the work to construct the metadata model for data management software, as well as the first results as harvested from a selection of the JERICO network partners. For the metadata model it has been very important to make the link to FAIR and make sure that the softwares metadata allows the user to determine if and how the software can be used.

Another important lesson is that it turns out to be not that obvious for researchers and data managers to think of sharing their code. Not all partners have responded to the questionnaire, and also later, when the system was upgraded to sharing software via Github, this still triggered only (a small) part of the community. It is recommended to include in future projects the demand for sharing any software within the partnership to fulfil the Open Science requirements in the coastal research domain.

As a potential next step, to further refine the proposed metadata schema, it will be essential to incorporate the upcoming outcomes of the OSTrails project (2024-2027), particularly regarding the representation of Data Management Plans (DMPs) within the resource catalogue and the mechanisms for conducting FAIR assessments which are represented above by the use case “Integration into Automatic Processes". The insights gained from OSTrails will enable a more nuanced understanding of how DMPs can be effectively integrated and represented within metadata frameworks, ensuring that the schema not only facilitates current data management and sharing practices but also aligns with future advancements in FAIR assessments and DMP integration. This adaptive approach ensures that the metadata schema remains relevant and effective in promoting interoperability, discoverability, and the seamless exchange of research outputs within the JERICO community and beyond, aligning with broader efforts to advance the principles of Open Science and FAIR data management.

8. REFERENCES

[1] Documentation of JERICO-RI e-infrastructure and capabilities (D7.6)
[2] ENVRI-FAIR Knowledge Base for RI Service Interoperation and Competence (D7.3)
[3] Code Metadata:

- https://github.com/codemeta/codemeta/blob/master/codemeta.jsonld
- https://codemeta.github.io/terms/

Appendix A

The table presents an overview of the software survey among JERICO partners, highlighting key details of each software tool. Initially, it offered extensive information, including the software's name, description, thematic area, maturity level, applicable platforms or instruments, programming language, current repository status, access details and preferred sharing methods, providing a comprehensive snapshot of the software landscape within the JERICO consortium.

INSTITUTION	Name	Software Theme	Maturity level	Platform (if applicable)	Programming Language	Link to online repository
SOCIB	Glider Toolbox	processing	community use	Glider	matlab, octave	https://github.com/socib/glider_toolbox
SOCIB	Leaflet.TimeDimension	viewer	global use		javascript	https://github.com/socib/Leaflet.TimeDimension
SOCIB	SOCIB Data REST API examples	data access	single developer		python, jupyter notebook, REST	https://github.com/socib/API_examples
SOCIB	SOCIB HF-Radar Automatic Report Generator	quality assessment report	internal use in organisation	HFR	python, octave	https://github.com/socib/HFRadarReports
SOCIB	NC2KML	format conversion	single developer		java	https://github.com/socib/NC2KML
UPC	SWE Bridge	driver	internal use in organisation	All platforms and instruments	C	https://bitbucket.org/swebridgedevelopment/swebridge
UPC	CSV2SOS	format conversion	single developer		python3	
UPC	M3	format conversion	single developer		python3	https://gitlab.emso.eu/Martinez/m3

hereon	GN4	processing	single developer	FerryBox, HFR, glider, Underwaternodes	java	https://gitlab.hzdr.de/gisbert.breitbach/gn4
FAMRI	CTD correction toolbox	QC	internal use in organisation	CTD	matlab	
FAMRI	ADCP correction toolbox	QC	internal use in organisation	Moorings	matlab	
CNR-ISMAR	HFR_Node__Centralized_Processing	format conversion	community use	High Frequency Radar	Matlab	https://github.com/LorenzoCorgnati/HFR_Node__Centralized_Processing
CNR-ISMAR	HFR_Node_tools	format conversion	community use	High Frequency Radar	Matlab	https://github.com/LorenzoCorgnati/HFR_Node_tools
CNR-ISMAR	HFR_Node__Historical_Data_Processing	format conversion	community use	High Frequency Radar	Matlab	https://github.com/LorenzoCorgnati/HFR_Node__Historical_Data_Processing
CNR-ISMAR	HFR_Node__REP_Temporal_Aggregation	format conversion	community use	High Frequency Radar	Matlab	https://github.com/LorenzoCorgnati/HFR_Node__REP_Temporal_Aggregation
CNR-ISMAR	HFR_Node__SDC_Processing	format conversion	community use	High Frequency Radar	Matlab	https://github.com/LorenzoCorgnati/HFR_Node__SDC_Processing
VLIZ	LifeWatch-EMODnet-Biology-QC-tool	quality assessment report	community use		R	https://github.com/EMODnet/EMODnetBiocheck https://rshiny.lifewatch.be/BioCheck/
VLIZ	worrms: WoRMS R package	processing	community use		R	https://github.com/ropensci/worrms
VLIZ	pyworms: WoRMS python package	processing	community use		Python	https://github.com/iobis/pyworms

VLIZ	mregions: MarineRegions R package	processing	community use		R	https://github.com/ropensci/mregions
VLIZ	eurobis: R package	processing	internal use in organisation		R	https://github.com/lifewatch/eurobis
VLIZ is hosting and using IPT, software itself developed by GBIF.	Integrated Publishing Toolkit (IPT)	format conversion	global use		Java	https://github.com/gbif/ipt
AZTI	JRadar Toolbox	format conversion	community use	High Frequency Radar	Java	https://github.com/Fundacion-AZTI/JRadar
SOCIB	HOORT: HFR Online Outage Reporting Tool	quality assessment report	community use	High Frequency Radar	Javascript	https://hoort.hfrnode.eu/
SOCIB	API-HOORT: HFR Outage Online Reporting Tool API	data access	internal use in organisation	High Frequency Radar	Python	https://hoort-api.priv.socib.es/api/outages
SOCIB	Eur_HFR_Node_API: European High-Frequency Radar Node API	data access	internal use in organisation	High Frequency Radar	Python	https://api.hfrnode.eu/

Appendix B

```
{
  "@version": 0.1,
  "@context": {
    "@vocab": "http://schema.org/",
    "schema": "http://schema.org/",
    "swo": "http://www.ebi.ac.uk/swo/",
    "iao": "http://purl.obolibrary.org/obo/",
    "edam": "http://edamontology.org/formats",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "required": "http://schema.org/required"
  },
  "@type": "SoftwareApplication",
  "@id": {
     "type": "@id",
     "description": "required @id persistent identifier"
  }
  "name": {
     "type": "schema:name",
     "description": "Your Software Name"
  },
  "description":  {
     "type": "schema:description",
     "description": "A brief description of your software."
    },
  "url": {
    "id": "https://www.yoursoftware.com",
    "type": "schema:URL",
    "format": "uri",
    "description": "The official URL of your software."
  },
  "keywords":  {
      "type": "schema:keywords",
      "container": "@list",
      "list": ["keyword1, keyword2, keyword3"]
  },
  "released": {
      "type": "swo:SWO_9000068",
      "description": "Official released date"
  },
  "domain": {
```

```
      "type": "schema:applicationCategory",
      "description": "Application category as for example science."
  },
  "downloadUrl": {
      "type": "swo:SWO_0000046",
      "description": "The URL for downloading the software."
  },
  "license": {
    "type": "swo:SWO_0000002",
    "description": "The URL to the software's licensing information."
  },
  "citation":  {
    "type": "schema:URL",
    "description": "Author Name et al. (2023). Your Software Name,
v1.0.0. Publisher. DOI: 10.12345/software-doi"
  },
  "author": [
    {
      "type": "swo:SWO_0000394",
      "description": "One of the authors of the software."
    },
    {
      "type": "swo:SWO_0000394",
      "description": "Another author of the software."
    }
  ],
  "publisher": [
    {
      "type": "swo:SWO_0004004",
      "description": "One of the authors of the software."
    },
    {
      "type": "swo:SWO_0004004",
      "description": "Another author of the software."
    }
  ],
  "supportUrl": {
      "type": "swo:SWO_9000070",
      "description": "The URL for downloading the software."
  },
  "documentation": [
    {
      "type": "swo:SWO_9000043",
      "description": "Associated documentation."
    },
```

```
    {
      "type": "swo:SWO_9000043",
      "description": "Another associated documentation."
    }
  ],
  "version": {
    "type": "swo:SWO_0004000",
    "description": "Version of software."
  },
  "code": {
    "id": "https://github.com/yourusername/yoursoftware",
    "type": "swo:SWO_0000001",
    "description": "The URL of the software's source code repository."
  },
  "dependencies":  {
    "type": "schema:dependencies",
    "container": "@list",
    "list": ["Dependency1, Dependency2"]
  },
  "language": {
    "type": "iao:IAO_0000025",
    "description": "Programming language"
  },
  "Input": {
    "type": "edam:format_1915",
    "description": "Data input"
  },
  "Output": {
    "type": "edam:format_1915",
    "description": "Data output"
  },
  "os": {
    "type": "swo:SWO_0000021",
    "description": "OS or multi-platform"
  }
}
```