



Report: Community workshop on Environmental model code of long- term value

NERC Environmental Data Services and Software Sustainability Institute

Michael Tso, Kit Macleod, Jennifer Roebuck, Philip Trembath, and Gordon Blair

01.06.2024



UK Centre for
Ecology & Hydrology

Contents

1.	Summary	3
	Recommendations.....	5
2.	Background and aims of the workshop	5
3.	Registrants and their experience	6
4.	Format of the workshop	12
	4.1 At a glance	12
	4.2 Keynote talk.....	12
	4.3 Lightning talks	12
	4.4 Breakout sessions and panel discussion	15
5.	Workshop findings	16
	5.1 FAIR requirements for environmental model codes of long-term value	16
	5.2 Fostering FAIR requirements for environmental model codes of long-term value (i.e., inputs).....	20
	5.3 Final comments	24
6.	Recommendations.....	25
	6.1 General principles.....	25
	6.2 Environmental modellers	28
	6.3 Funders	28
	6.4 NERC EDS.....	29
7.	Next steps	29
8.	Acknowledgments	30
9.	Appendix 1: Agenda	31
10.	Appendix 2: Resources and further reading	33
	10.1 References	33
	10.2 Video	33
	10.3 Tools	33
	10.4 Case studies and articles.....	34
	10.5 Communities	34

1. Summary

Software is increasingly essential to research activities, from data collection to its processing and storage, and its publication and use. It underpins other scholarly research output and can be used, reused, re-examined, and further developed (Katz and Chue Hong 2024). Recognising the role of research software in environmental science, this report covers a workshop on environmental model code of long-term value run by The UK Centre for Ecology & Hydrology (UKCEH) on 21st November 2023.

The main goal of the Natural Environment Research Council Environmental Data Services (NERC-EDS) is to “ensure that environmental data are made available, accessible and re-usable for the long-term in order to fully realise their value”. Increasingly, these datasets are modelled outputs, and the models that generated them are valuable resources that have the potential to be re-used for further research and wider studies. Therefore, there is a need to ensure the software code that comprises and runs models to generate datasets are also of long-term value.

The one-day, hybrid workshop aimed to bring together NERC-EDS staff and UK environmental modellers to scope the principles and inputs needed to advance the long-term value of environmental model code. It was sponsored by a Software Sustainability Institute (SSI) fellowship grant, and 38 UK-based environmental modellers, data stewards, research software engineers and other researchers and professionals attended. The participants offered inputs through lightning talks, breakout groups, and open discussions.

In the first breakout session, the participants brainstormed and discussed *What are the FAIR¹ requirements (e.g. users, funders, publishers) for environmental model codes of long-term value?* In the second session the participants brainstormed and discussed *How may we foster the developments of environmental model codes of long-term value (i.e., inputs)?* These were grouped into existing resources, resources that are required or need increased, and greater recognition of when best practices are implemented.

¹ The FAIR principles provided internationally recognised guidelines to improve the Findability, Accessibility, Interoperability, and Reuse of digital assets. See more at <https://www.go-fair.org/fair-principles/>.

As a group, the workshop attendees agreed that greater effort is needed to ensure that as a community we produce environmental model code of long-term value. This is because science and applied analysis is increasingly reliant on research software, and that these resources are FAIR. We identified five challenges to delivering model code of long-term value. These challenges span planning and understanding requirements, writing code, maintaining code, using other people's code, and sharing code. In terms of **planning and understanding requirements**, challenges include planning and managing code after project end and understanding user and funder requirements. For **writing code**, challenges include time constraints for thorough testing and developing documentation, adapting to diverse domains with varying best practices, and dealing with limited data availability. **Maintaining code** poses challenges in ensuring its future compatibility, especially when dealing with dependency updates that may introduce bugs. When **using other people's code**, challenges arise from insufficient documentation or comments that fail to explain the rationale behind certain decisions. Finally, challenges related to **sharing code** include bridging gaps between research software engineers using different programming languages, making code accessible to non-experts, maintaining consistency between scientific repositories and version control, and ensuring all model code artifacts are available for reproducibility.

Additionally, the key themes (and needs) raised during the workshop to address the above challenges were:

- **User guidance**
The workshop participants (both modellers and data centres) felt there was a lack of guidance material for best practices of managing and publishing model code. Case study examples of good practice would also be helpful.
- **Community building and stakeholder engagement**
Work with existing modellers and wider environmental data scientists to develop a broad community to share best practices and experiences.
- **Technical support and training**
Research software engineers, data scientists, data stewards, modellers and data engineers championing and supporting software sustainability are key to prolonging the value of model codes and providing training to wider end-users.
- **Assets commons**
A more holistic view of FAIR (**F**indable, **A**ccessible, **I**nteroperable, **R**eusable) for research objects and assets is needed. This includes data, models, methods and instruments/platforms.
- **Greater resources and funding to address these needs**
Like any infrastructure, codes need investment. This includes funding to maintain, archive, and document codes after the end of projects. Funders should not only encourage new software, but fund the sustainable development of key software projects used by different communities.

Recommendations

We have provided recommendations to address the five challenges to delivering model code of long-term value. These specific recommendations are provided in [section 6](#).

In addition, we highlight the following to accelerate changes to prolong the long-term value of model codes.

- We recommend that **guidance material** for best practice for model code management is produced. This should be coordinated by the NERC EDS, in consultation and collaboration with the environmental modelling community.
- Alongside any guidance, we recommend that **real-world examples** of good/best practice be published. For example, recommendations in Garijo et al. (2022) are provided as nine separate policies or statements, each presented below with an explanation as to why we recommend the practice, what the practice describes, and specific considerations to take into account.
- **Expand data stewardship support** of model codes to those that are used in multiple projects.
- We encourage the **involvement of research software and data professionals** such as research software engineers, data scientists, data stewards, and modellers in all relevant project and work plans. Importantly, they should be included from the very beginning i.e. the proposal writing stage.
- We recommend NERC-EDS continue the work of scoping and development of an **asset commons** (Siddorn et al., 2023) and, in so doing, learn from the experiences of other organisations/initiatives using this approach.

2. Background and aims of the workshop

The main goal of NERC data centres is to “ensure that environmental data are made available, accessible and re-usable for the long-term in order to fully realise their value”. Increasingly, environmental artifacts include not only data collected in the field or samples analysed in the laboratory, but also model outputs that are generated by model code. Model code also plays a vital role in data analysis, understanding and visualisation, and in decision making.

For the purposes of this workshop, we focus on model codes that are used to directly generate research outputs (e.g., process models, statistical models, and machine learning models), which are of more immediate relevance to NERC EDS. However, the general principles developed can be applied to all research software (e.g., software packages, digital research platforms).

The one-day workshop was hosted on behalf of NERC Environmental Data Service (NERC-EDS) Special Interest Group (SIG) on Models and was sponsored by a Software Sustainability Institute fellowship grant. It brought together a community of practice to develop and share principles and practices to promote the long-term value of environmental model codes, and identified potential tasks for funders, data centres, environmental modellers, and research software engineers to focus on, to foster such practices.

3. Registrants and their experience

Thirty-eight people registered for the workshop, and as part of the registration they were asked short questions so we could gain a better understanding of their area of expertise and their coding experience and challenges. There was a good mix of modellers, data managers or data stewards, data scientists, and research software engineers (RSE) (Figure 1). A wide range of environmental science sub-disciplines were represented, as shown in Figure 2 . Figure 3 shows a word cloud from the response for “What makes you interested in attending this workshop?” A lot of the answers centred around the use of environmental models and code within the community and following best practices.

The registrants were also asked a multiple-choice question of “What is your model code written in and how often?”. The three most popular answers were Python, R and Jupyter Notebooks, whereas no one selected GMT (Figure 4). In terms of publishing code, GitHub proved to be the most popular option among the registrants (Figure 5).



Figure 1 Word cloud generated from job titles of the workshop participants.

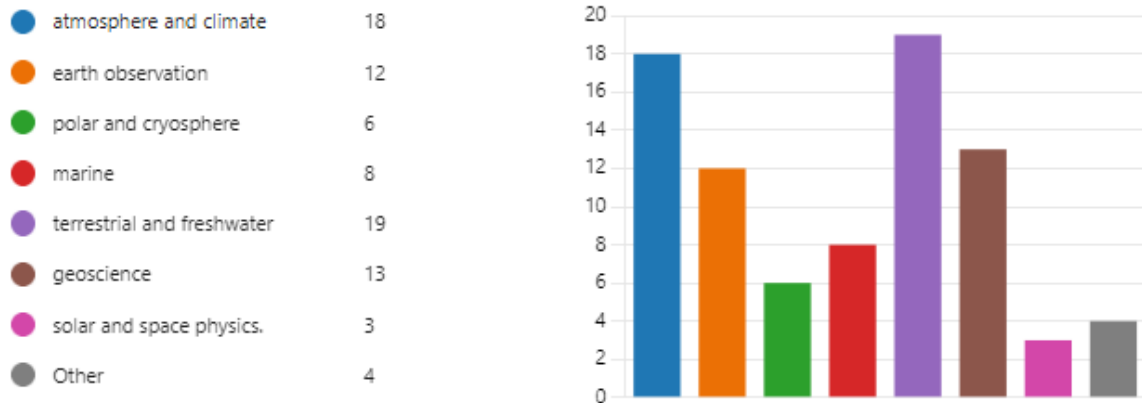


Figure 2 Breakdown of environmental science sub-disciplines represented in the workshop. Multiple users selected more than one and some respondents put “interdisciplinary” in the free text box.

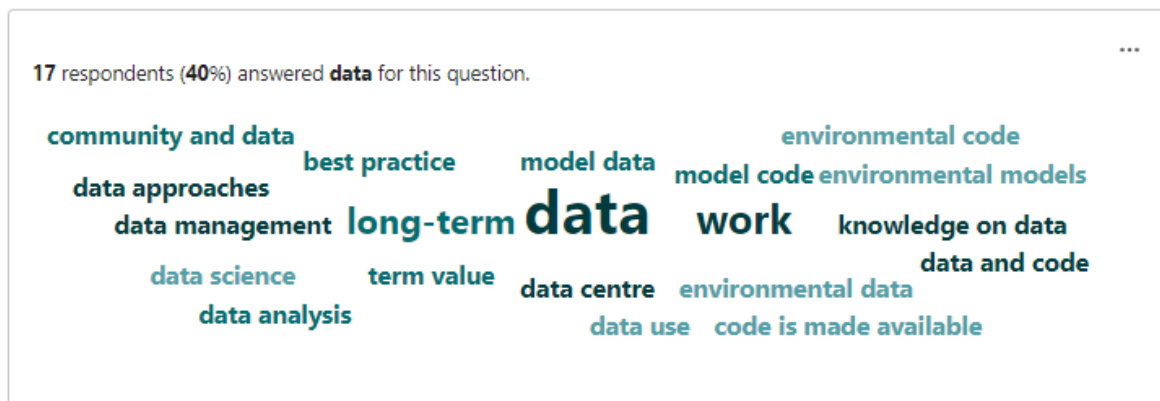


Figure 3 Word cloud results for “What makes you interested in attending this workshop?” in the workshop registration form.

15. What is your model code written in, and how often?

[More Details](#)

■ Frequently ■ Sometimes ■ Never

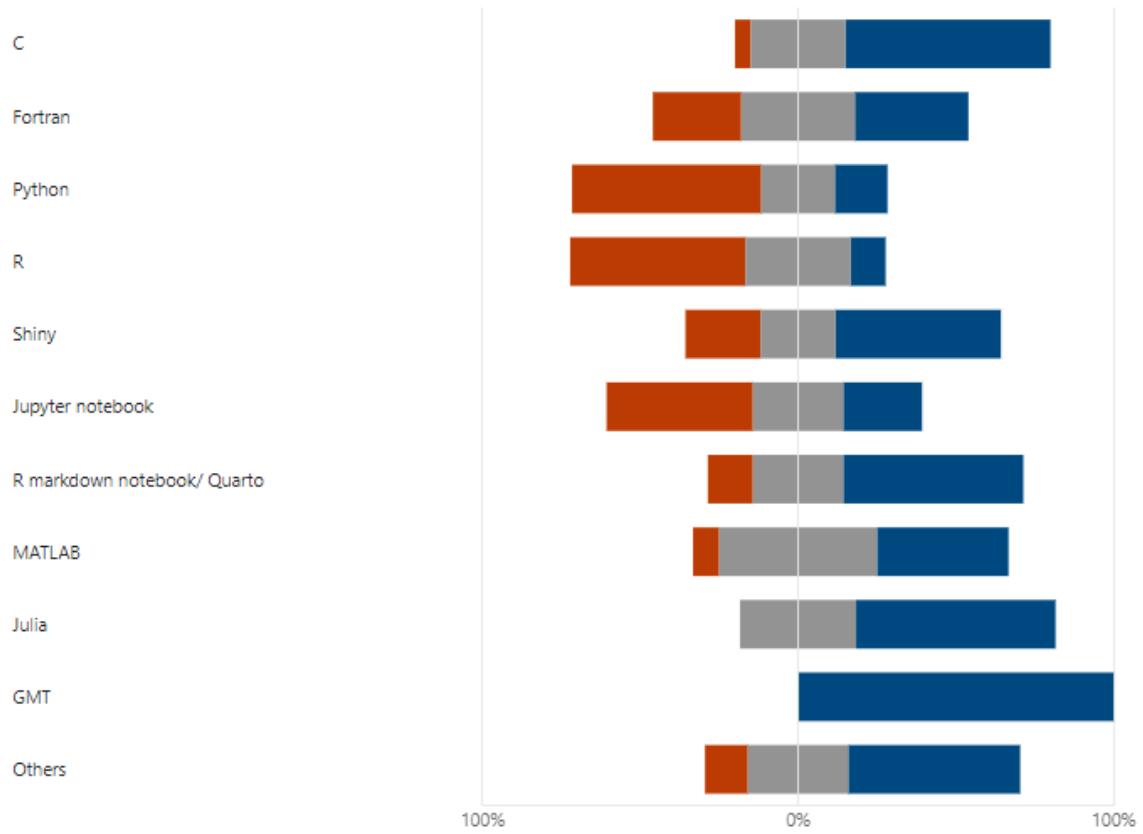


Figure 4 Coding languages used by workshop participants

16. Where have you published your model codes?

[More Details](#)

RO-Hub	1
Research Elements (Elsevier)	0
GitHub	27
GitLab	11
Zenodo	13
Hydroshare	0
A NERC data centre (i.e. BODC, ...)	5
A university repository	4
RO-crate	0
FigShare	1
A domain-specific repository	2
A software repo (e.g. CRAN, pip)	7
Met Office Science Repository S...	3
CSDMS model repository	1
Other	2

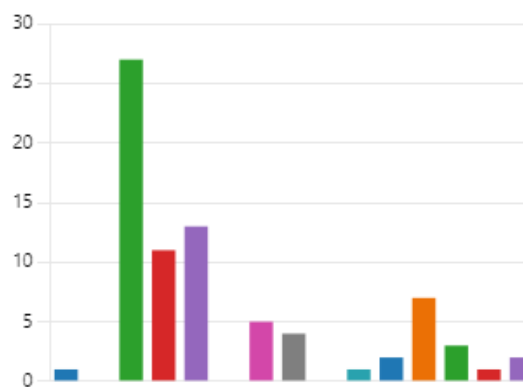


Figure 5 Many participants have published model codes in GitHub or GitLab, followed by Zenodo and language-specific repositories.

Many registrants expressed enthusiasm on the topic of environmental model code of long-term value in their registration forms. Some have worked in this space for a while, and others have recently started exploring it. Several NERC data centres, such as the National Geoscience Data Centre (NGDC), seem to have frameworks being developed around the concepts of FAIR software and are looking into ways for wider implementation.

Some of the specific challenges on prolonging the value of the model code highlighted by the registrants were: (i.) burden to maintain, test, and document code, (ii.) reusability of components, and (iii.) reusability of code.

Burden to maintain, test, and document code

- “Time to properly test and document stuff.”
- “Keeping the code up-to-date has been pointed by few people as the main challenges. Perhaps more rigorous use of sandboxing environment and package management software can help here.”
 - “One of the major challenges is to keep the code up to date in line with the programming languages developments. For example, to manage R packages dependencies in one's own package.”

- "Keeping software up to date, particularly ensuring machine learning models remain secure. For example, recently having to update the H2O library in a Conda environment, because the version on Conda is outdated, and uses an unsupported version of log4j, from 2019."
- "Changes in software (e.g., R packages deprecation)."
- "Keeping up with ever updating software and code."

Reusability of components

- "how to create a subroutine that can be easily implemented to other models."
- "Making the code accessible to users of other languages or non-coders. Notebooks can be helpful here."
- "We collaborate across multiple projects and institutions with code contributors inside and out of our organization. It is therefore important to us that a standard of coding documentation is adopted across multiple user backgrounds."
- "Developing code in a way that's flexible to future changes, but fast enough to deliver immediate project requirements. An understanding of how to design libraries to support multiple analyses, extract models with clear interfaces, or continually review and incorporate data/model updates."

Reusability of code

- "Legacy code with no documentation or meaningful comments."
- "Data availability and openness, as well as data accretion and changes to collection protocols."
- "Also, ensuring data used to train models is archived, or recorded in a database. The provenance of my organization's data can at least be recorded in our database. However, while it hasn't happened yet to me, it is entirely possible I could be asked to train a model on a dataset, which would only be made available for the duration of a project (for example an external agencies data). This would make it impossible for future researchers to reproduce or improve model output."

Others

- "Reproducibility often seems to be the biggest hurdle. Archiving code is relatively straightforward (e.g., via Zenodo's link with GitHub), but being able to re-create the computational environment to re-run that code years after it was archived can be a real challenge, even for language with "robust" package management."
- "Standardisation of approaches to making things FAIR, especially cross-discipline, for a variety of users and developers over the long term. "
- "Infrastructure"

- "Models being updated so regularly that 'snapshots' of code may be given DOIs or made accessible through Zenodo, but they are already out of date. In-team collaboration on code is not necessarily done effectively using versioning software like Git (and sometimes not even through cloud storage)."

Finally, here are some of the registrants' responses when they were asked about their experiences or "hopes and dreams" for environmental models:

- "Science requires predictions as well as observations."
- "Modular reusable code bases for environmental modelling"
- "I would like the modelling community to engage the documentation effort with enthusiasm."
- "Never-ending dream of all software, not only environmental models, to be open-source, and user-friendly!"
- "It would be great if there was some way to automatically check which software packages in a Git repository were no longer supported, so you could see what needed updating immediately."
- "Also, a sandboxed environment to use older packages (for example when checking results from older work), so the security concerns of using older packages could be made safer."
- "The nature of environmental modelling especially for applied use cases is changing by researchers (let alone use by the rise of ClimateTech ResilienceTech companies, and the environmental modelling by Microsoft, GEE etc.). In addition to having a standalone executable, often modelling involves a series of pipeline components/workflows in batch or near real time. Many other changes including serverless and the increasing importance of green software engineering."

4. Format of the workshop

4.1 At a glance

This was a one-day, hybrid workshop that focused on connecting participants and stimulating discussions. The Sli.do website was used for questions in the main session to ensure that both online and in-person participants could ask questions and the questions that got the most interest will be answered. See the workshop agenda in [Appendix 1: Agenda](#).

4.2 Keynote talk

Prof. Chris Jewell from Lancaster University gave a keynote talk on the Generalised Epidemic Model (GEM) and the experience of using it in a reproducible manner to generate automated daily reports in a useful format for central and local governments during the COVID-19 pandemic.

The statistical model setups for both hospital and national-level COVID-19 transmissions were introduced. Key model outputs include identifying transmission hotspots and generating spatial maps of the basic reproduction number of the virus (i.e., R number). It comprises of running a transmission model and identifying its parameters using Bayesian methods, which is repeated daily to provide the most up-to-date estimates.

The various challenges and the solutions responding to them were discussed. The roles of data format and pre-processing, a reproducible workflow and data pipeline, automated report generation and cyberinfrastructure within a high-end computing context were highlighted. A new high-level domain-specific modelling language (DSML, based on Python) for epidemic modelling was introduced, which would allow the GEM to adapt quickly to a different epidemic and with different choices of tools (e.g., Bayesian modelling software).

4.3 Lightning talks

The lightning talks were designed to stimulate discussions in breakout groups and coffee breaks. The lightning talks allowed participants to briefly highlight some of their work and share their thinking on the workshop topic should they wish to do so.

Gordon Blair posed the question “Where modelling should go?”. Modelling is often viewed as a second-class citizen to data. FAIR is mostly thought about in the context of data, but modelling is seen as not as important. He noted that there’s still lots of work to do with FAIR data. Commons technology has been proposed to foster access and reuse data. It should go beyond data commons and extend to an asset commons. The [Models in the cloud](#) project was mentioned as a starting point to having FAIR models—in particular, accessible models, that are ready to be executable on-demand in the cloud. It uses a contemporary cloud software, stacks. It explores ways of containerizing models effectively, using the WRF model as an example. Once the model is ready to be executed on-demand, then it can be catalogued, described, discovered, and executed.

Ed Rowe shared his experience and best practices in designing dynamic process models, which often involves various flows and stocks. It is common to start with sketches and sometimes it involves translating from one language to another. Some lessons learned included the efficient use of naming conventions for multidimensional arrays and the use of Continuous System Modelling Protocol proposed by IBM. He ended by encouraging us not to be afraid of new ideas and to keep looking for ways to communicate with and learn from coders and non-coders.

Rich Ellis talked about the new [LTLS Freshwater Ecosystems](#) (LTLS-FE) project which investigates river ecology under future scenarios for the entire UK. It involves lots of closed source codes and it cannot have identifiable datasets. It will also create a large amount of output data, in NetCDF format, and the project team is starting to think about a strategy to handle this challenge.

Alejandro Coca-Castro highlighted the Environmental Data Science (EDS) book as an excellent online resource, which contains FAIR Computational Notebooks (<https://edsbook.org>). All the notebooks are discoverable and FAIR, all with metadata, badges, and are linked to RoHub (<https://rohub.org>) as research objects (RO) with DOIs. All the resources on RoHub are reproducible with binder and JupyterHub, shareable as JupyterBook, and use PANGEO open-source tools. They can also be exported to RO-crate, making them completely interoperable.

Tom Russell shared his work on water resources modelling in England and Wales as a research software engineer. The work models future climate conditions transfers using systematic systems and modelling to incorporate climate scenarios, hydrology, streamflow in water resources systems. [DAFNI](#) was also highlighted as a resource. Specifically, the following workflow was discussed: “Source code >> container registry (STFC Harwell) (data parameters, outputs) >> DAFNI workflows (visualization)”.

Charlotte Pascoe shared their work on capturing information of experiments in CMIP6 (<https://es-doc.org/cmip6/>, Figure 6). In the previous round of the project, CMIP5, quite a lot of documentation was mandated, and the modelling groups hated it. Now analysis tools exist, the requirements are more lenient, there is less burden on modellers, but the coverage of documentation is sparser. In the future, CMIP7 is aiming to get the balance right for good coverage and minimum burden.

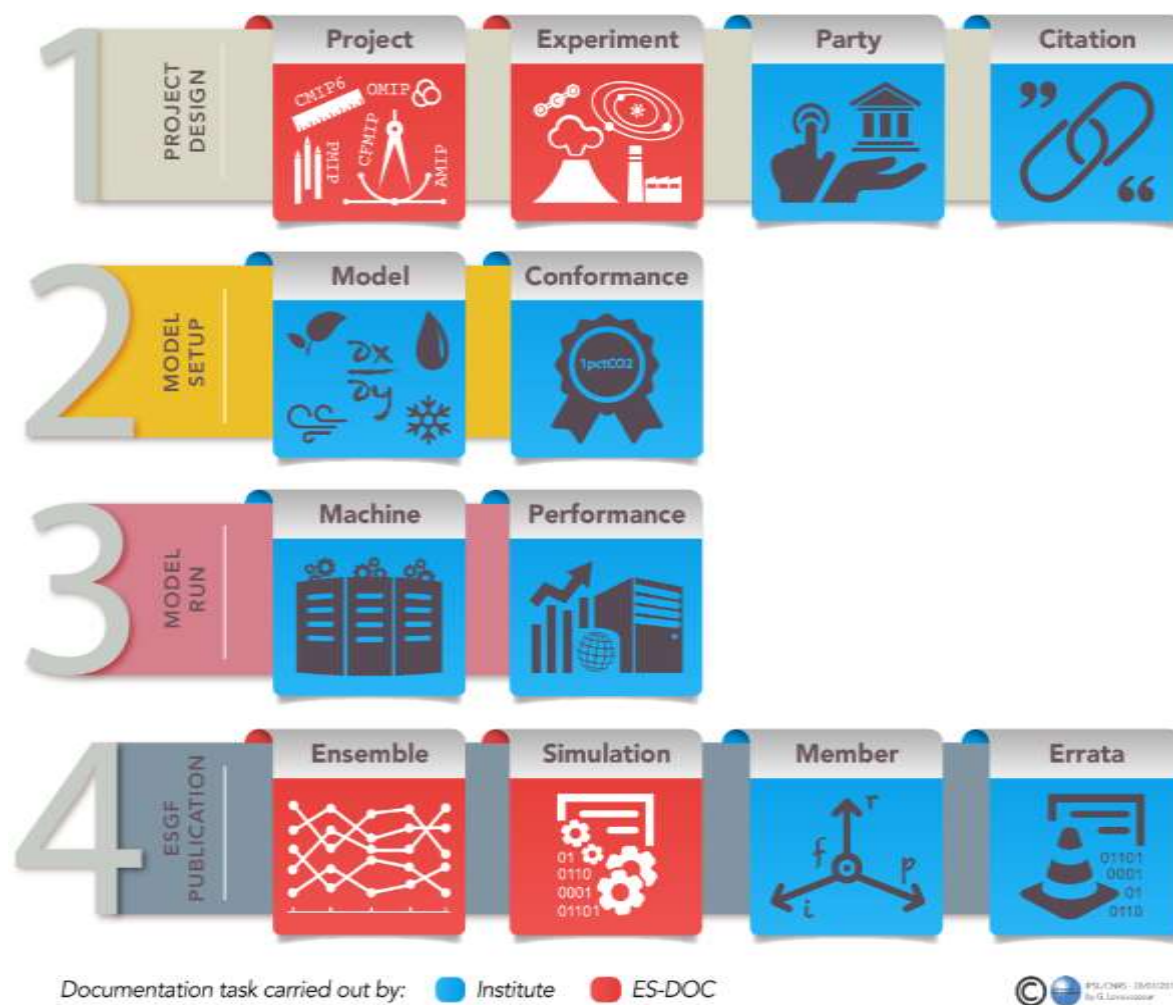


Figure 6 Infographic showing the documentation tasks in CMIP6 modelling studies.

Ana Couto shared their experience, progress and challenges of working on the [Cumulative Effects Framework](#) (CEF) project, which involves linking different modelling tools. The three key requirements for developing CEF were; (i.) A data library, knowledge base, and data store; (ii.) R package containing functions to run each of the modelling tools, link them together in feasible combinations, and perform a project level or cumulative assessment; (iii.) A user interface that allows non-technical users to generate predicted impacts at a population level for both individual projects and cumulative assessments, with a clear audit trail to provide transparency and reproducibility.

Kit Macleod shared their experience in a wide range of software projects, highlighting the longer term (5-10 years) needs of users and funders. His talk also gave many examples of existing research community principles and approaches. He finished with a tribute to [Dr. Peter Fox](#), who argued for a focus on *cultural* activity of *standards adoption*, and on *technical* activity of *standards creation*.

4.4 Breakout sessions and panel discussion

During the workshop there were two breakout sessions and a panel discussion (Figure 7). In small groups the participants discussed four questions over two breakout sessions with a panel discussion at the end. The first session (Session 1) focused on FAIR requirements for model codes and summarising them as principles, while the second session (Session 2) aimed to gather potential inputs that could help achieve outcomes that align with the principles. The outputs from these are described in further detail in Section 5.

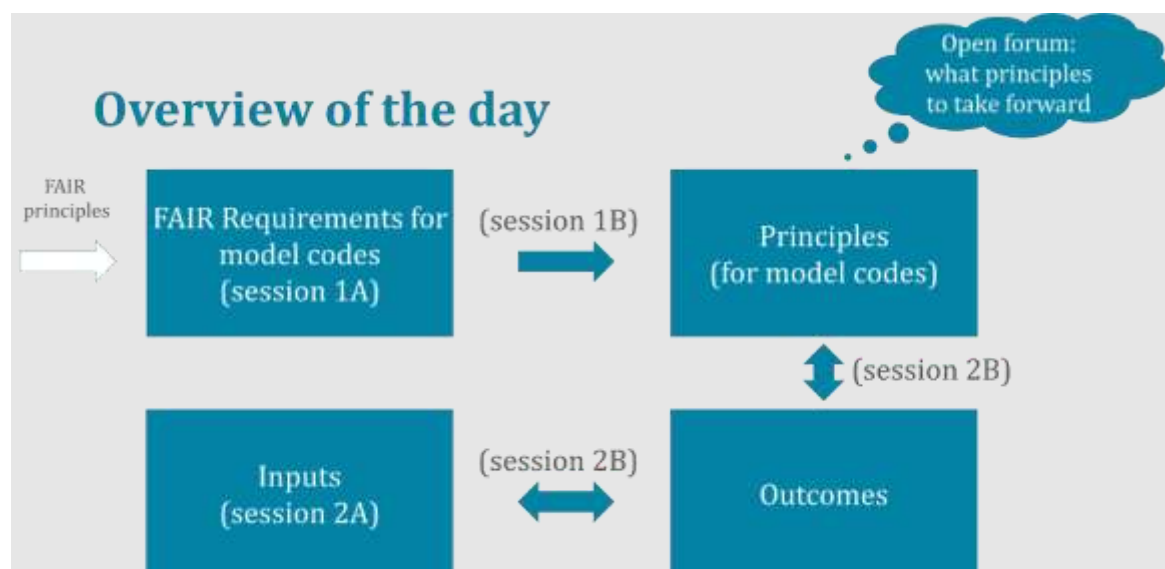


Figure 7 Conceptual diagram of the discussion sessions in the workshop.

5. Workshop findings

In session 1 the participants brainstormed and discussed What are the FAIR requirements (e.g. users, funders, publishers) for environmental model codes of long-term value? In the second session the participants brainstormed and discussed How may we foster the developments of environmental model codes of long-term value (i.e., inputs)? These were grouped into existing resources, resources that are required or need increased, and greater recognition of when best practices are implemented.

5.1 FAIR requirements for environmental model codes of long-term value

In the first breakout session the groups discussed the following question:
1. What are the FAIR requirements (e.g., users, funders, publishers) for environmental model codes of long-term value?

The groups recorded their thoughts on sheets using sticky notes (physical or digital versions depending on whether they were in-person or online). Figure 8 shows what these sheets looked like, and Table 1 is a summary of the participants' inputs.

Session 1A: What are the FAIR requirements (e.g. users, funders, publishers) for environmental model codes of long-term value?

(Group Name)

Drag and drop sticky notes

FAIR

Findable

Accessible

Interoperable

Reproducible

Repeatable, replicable, reproducible, reusable

Others

Comments

Figure 8 Screenshot of the activity sheet given to participants in session 1.

Table 1 Participants' input in session 1-- What are the FAIR requirements (e.g. users, funders, publishers) for environmental model codes of long-term value?

FAIR principle	Requirements
Findable	<ul style="list-style-type: none"> • Link to code from the research papers that used it. • Listing in searchable resources and have unique identifiers (DOI code). • Very important, or the success of the other requirements don't matter. • Publishers - Search engine optimisation. • Users - Finding the correct version. • Asset commons (also facilitates the other bits of FAIR). • Needs top-down requirements from funders. • Need the motivation, recognition that papers are not the only output. • Papers, blog posts, social media--used to find/keep up to date with models, following research groups. • Package registries: document code here. • Need to make it attractive to make models available--so that they are findable
Accessible	<ul style="list-style-type: none"> • Providing different available languages. • Pseudo code to describe the intention of code. • Consider the purpose of documentation. Who is the audience and who is the potential audience? • Can be value to both you and other users. • Users: Design and documentation for applications and interfaces. • Funders: can drive adoption by advocating openness. • Closed software - can only use if you have the funds. • Government specifications: container image. • Pin the package versions key to access/reproducibility.

FAIR principle	Requirements
Interoperable	<ul style="list-style-type: none"> • Users/modellers - Dockerisation? • Define lifetimes of model. • Machine readable metadata. • Standards. • "Complete" asset commons. • User / publisher: Avoiding or considering eventual vendor lock-in. • Metadata: Needs a system that recognises both human readable and machine readable. • CF Compliant model is the dream but requires lots of time and work. • GUI make it user friendly--don't have to worry behind the scenes computing and technical details.
Reproducible (repeatable, replicable, reproducible, reusable)	<ul style="list-style-type: none"> • Data codes should be modular and adaptable. • What is long term means? Maybe training needed? • Clear which packages and tools used. • Users/modellers - Dockerisation? • Define lifetime of model. • Versioning / state recording. • Automating processes that derive information. • Choice of programming language. • Use what is best for the problem - the issues come from not having well defined interfaces, documentation etc. • Resource to meet these requirements, e.g. data stewards • Code should include built-in common-sense checks. • Need to provide info on common errors--so that users can fix them. • Being able to change parameters easily. • Input data need to be 'next to the model'. • Documentation need resources. • A suggested framework. • Need input from community on what it should include. • Good metadata available with the model. • Need to be adaptable to new data/ new science questions. • Example model frameworks. • Detailed documentation needed but it is a burden time-sink.

FAIR principle	Requirements
Others	<ul style="list-style-type: none"> • Green computing
Comments	<ul style="list-style-type: none"> • How do we make people do this [FAIR] with models? • Funder requirement? Top-down? Community-led is better but it takes time. • Hire people for doing this. • What is the equivalent of data papers for models? • model catalogues like we have data catalogue. • Not everything needs to be curated

5.2 Fostering FAIR requirements for environmental model codes of long-term value (i.e., inputs)

In the second session the breakout groups considered the following question:

How may we foster the developments of environmental model codes of long-term value (i.e., inputs)?

Figure 9 shows the sheet that the participants used to record their thoughts, and Table 2 summarises their inputs.

Session 2A: How may we foster the developments of environmental model codes of long-term value (i.e. inputs)?

Resources (e.g. tools, guides, training materials) <i>Existing:</i> <i>Needs development:</i>	Opportunities	
Skills development and training	Challenges, risks, and barriers	
Recruitment	Recognition/ celebration	Others

Figure 9 Screenshot of the activity sheet given to participants in session 2.

Table 2 Participants' input in session 2--How may we foster the developments of environmental model codes of long-term value (i.e., inputs)?

Category	Items
Resources (existing)	<ul style="list-style-type: none"> • NCAS introduction to scientific computing • JASMIN and DataLabs - good documentation on how to setup and use • The Turing Way, EDS book • chatGPT • EDS book, PANGEO, Elixir: look at these for examples to build upon • Data stewardship wizard: with a bit modification could be used for documentation for models. • Data stewardship wizards (DSW) • PyOpensci/ROpensci--guides for developing maintainable open software. • 'Model' stewards embedded in science areas/groups - like data stewards in UKCEH
Resources (needs development)	<ul style="list-style-type: none"> • Investing in robust computational software tools • Presence of research software engineering staff and other support in institutes. • Time for continuous improvement and Assessment • Need a forum for e.g. peer-to-peer knowledge sharing • Scientific user groups or similar • Build on work with organisations that are already doing this. • UK Reproducibility network: they have guidance on producing open source code. • Define community standards

Category	Items
Skills, development and training	<ul style="list-style-type: none"> • The Carpentries training courses. • Providing training and professional development. Having a course when you are undergraduate, learning FAIR principles. • Writing code to be understood by others. • Learning GitHub-style sharing environment early in the career. • Keep it simple / engineering. • Must consider return on investment of training and development efforts. • Peer-to-peer training like in ELIXIR. • Investment in training course • Generate metadata as you go along no as an afterthought. • Legacy environmental code: hard to bring it in line with FAIR, e.g., moving WRF to the cloud. • Budgets for training existing staff - e.g., Turing courses. • Cloud computing expertise. • Turing/RSE/RAMS etc. • RES - RSE boundary spanning. • Learn from ML Ops.
Opportunities	<ul style="list-style-type: none"> • Collaboration, learning from each other, improving the code, bug. • Increasing awareness of existing model landscape across domains. • Increasing awareness of opportunities to collaborate (these could/should be funded). • FAIR friendly scientist to be a go between. • More research infrastructure roles.
Recruitment	<ul style="list-style-type: none"> • Hiring HPC staff and research software engineers to support researchers. • Community managers. • Need for more cross-discipline people, e.g., scientific domain specific RSEs? • Research software engineers--recruit ahead of time. • Technical writers. • Recruit at the start of project or well in advance.

Category	Items
Recognition/celebration	<ul style="list-style-type: none"> • Reproducibility awards, e.g., CODECHECK. • Innovation showcases. • Reproducible software should be valued the same as or more than a 'Nature' paper. • At conferences have 'best code' prize. • Include requirements within job description or promotion requirements. • Case studies of when models have been made FAIR - demonstrate benefits to scientist + community. • Citation metrics • Creating communities
Others	<ul style="list-style-type: none"> • Communication.
Challenges/Risks/	<ul style="list-style-type: none"> • Package and code not being updated. • Lack of funding. • Clarity of inter-departmental and general communication. • How do we enthuse people about FAIR4RS, software sustainability etc.? • How do we pitch training at the right level for people's skills/interests? • FAIR-OPEN discussion. • What service level agreements (SLA) provided by modeller/science? • Budgets for, and how long to maintain code? • Cloud vendor lock-in: hard to transfer code between different vendors, e.g. AWS, Google. • Cultural resistance to a new way of working.

5.3 Final comments

Here are some of the final comments raised during the panel discussions:

- Discussion like the ones in this workshop can help shape future NERC EDS commissioning.
- “Don't expect everyone to do everything”—while there are many best practices and steps can be taken, we should give clear, prioritized message to users and practitioners and be realistic that not everyone will do everything suggested at once.

- “Under a larger banner”: an example of the ELIXIR community in bioinformatics gives a focal point of resources and tools to advance FAIR practice.
- A combination of themes, examples, and requirements to push FAIR model code.
- What’s the next big ambitious model code (2nd gen of JULES, Met Office)? A new, exciting model code project may attract attention and effort.
- RSEs as go-betweens
- “Hybrid RSEs”, both general and embedded in science topics. 50% of their role should be about extending FAIR data practices to more generally assets or research objects, while the other 50% is about cultivating domain-specific capacity and community for FAIR software.

We acknowledge that each software language (e.g., R, Python, Fortran, JS, Go, Rust) and modelling community (e.g., agent-based modellers) often has its own best practices, as well as wider software best practices from classic books such as Clean Code. Several of the issues raised at the workshop had already been solved and implemented out with research software engineer community. A key area of work will be to help each modelling community to better learn from one another and adapt solutions to its applications.

6. Recommendations

6.1 General principles

These principles were suggested by the participants and are aggregated and grouped by the report authors. These recommendations are also mapped to the following categories of challenges:

- Planning and understanding requirements = P
- Writing code = W
- Using other people’s code = U
- Maintaining code = M
- Sharing code = S

Project and software life cycle management, governance, and standards

1. Always start a project with a consideration of the FAIR requirements. [P]
2. Compatible with other data formats, software and workflows. [S, U, W]
3. Understand wider user requirements. [P]
4. Setting expectations and enabling FAIR principles from the start of project. [P]
5. Legacy planning of model code. Make plans and provisions to maintain the code long term (i.e., beyond end of project). [P, M]
6. Model codes should be stored in a way that they remain retrievable and understandable over time. [W, M, S]

7. Licensing and service-level agreement should be in place, and they dictate how users use them. [P, U, S]
8. Guidelines and common standards for models. Follow them where applicable. [W]

Coding practice

10. Use version control. [W]
11. Automate your processes. [W]
12. Compatible with other data formats, software and workflows. [W, S]
13. Require open and straightforward access. [P, S]
14. Ensure your stakeholders can access the model, or its outputs. [P, W, S]
15. Avoid vendor lock-in and aim to be as open source as you can. [P, W, S]
16. Follow applicable standards. [P, W]

Documentation and long-term usability

17. Model codes should be stored in a way that they remain retrievable and understandable over time. [M]
18. Should be well documented with clear and accessible metadata. [W, S]
19. Clear, concise, documentation, both in code, and pdf/a compliant pdf. [P, M, S]
20. Document code for end users. [M, S]
21. Ensure code is easily maintainable in the long-term i.e., it be easily re-used by others long beyond the end of the project. [P, S]

People, skills, and culture

22. Building capacity for people, skills, and communities. [all]
23. Working in collaboration - valuing every role. [all]

Many of these principles are similar to those proposed as FAIR for Research Software (FAIR4RS) principles last year (Barker et al., 2022), convened by Research Data Alliance's FAIR4RS working group. It was only briefly outlined in one of the lightning talks, but the participants arrived at very similar conclusions. The FAIR4RS principles are listed below:

Findability: Software, and its associated metadata, is easy for both humans and machines to find.
F1. Software is assigned a globally unique and persistent identifier.
F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
F1.2. Different versions of the software are assigned distinct identifiers.
F2. Software is described with rich metadata.


F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. Metadata are FAIR, searchable and indexable.
Accessibility: Software, and its metadata, is retrievable via standardised protocols.
A1. Software is retrievable by its identifier using a standardised communications protocol.
A1.1. The protocol is open, free, and universally implementable.
A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the software is no longer available.
Interoperability: Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.
I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects.
Reuse: Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).
R1. Software is described with a plurality of accurate and relevant attributes.
R1.1. Software is given a clear and accessible license.
R1.2. Software is associated with detailed provenance.
R2. Software includes qualified references to other software.
R3. Software meets domain-relevant community standards.

FAIR4RS Principles

Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A-I, Martinez, C., Psomopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., Honeyman, T., et al. (2022). FAIR Principles for Research Software version 1.0. (FAIR4RS Principles v1.0). Research Data Alliance. DOI: [10.15497/RDA00068](https://doi.org/10.15497/RDA00068)

- **Findable:** Software, and its associated metadata, is easy for both humans and machines to find.
- **Accessible:** Software, and its metadata, is retrievable via standardized protocols.
- **Interoperable:** *Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.*
- **Reusable:** *Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).*

(key differences from FAIR data principles in *italics*)



<https://doi.org/10.5281/zenodo.7613361>

9

Figure 10 Summary slide highlighting the key differences between FAIR4RS principles vs FAIR data principles.

6.2 Environmental modellers

- Adopting FAIR4RS Principles
- Build a vibrant community to share best practice and ideas and feedback key findings to NERC, not only existing as standalone network but to support connected network of environmental modeller/engineers/data scientist communities.
- Raise awareness of the critical roles that environmental model code provides with funders, researchers, and end users of research.
- We acknowledge that some environmental modellers are research software engineers, while others work more as researchers. Skill sharing and collaboration of different roles are encouraged.
- Linked to this awareness is the need to increase capacity in digital support for environmental research. How to increase this capacity requires diverse solutions, from local low-cost sharing to large scale investment, like the individual and organisations it supports. The ability of multiple actors in society to deliver sustainable development under environmental change is dependent on increasing this capacity.

6.3 Funders

- Adopt and promote FAIR4RS Principles in funding calls.
- Funding projects to prolong the life cycle and interoperability of key legacy codes.
- Fund new, ambitious model code projects.

- Encourage the funding of research software engineers, data engineers, data stewards, and data scientist in funding calls. Provide funding for post-project code maintenance.
- Include FAIR digital assets (including model codes) in its future scientific and digital strategies.

6.4 NERC EDS

- Promoting FAIR4RS Principles in internal training and external communications
- Incorporating FAIR4RS Principles in future EDS enhancement projects
- Facilitate further sessions to share best practices.
- Start a working group to develop guidelines for data centres to support FAIR model codes.
- Develop unified approach in NERC-EDS for how we handle model projects, meeting long-term storage requirements for both model code and outputs.
- Engaging a community to develop metadata standards.
- Work towards a digital asset commons approach to increase findability.
- Currently, data stewards support model codes and software packages developed within a project. This can be extended to software that underpins multiple projects. A stronger emphasis on post-project maintenance and life cycle planning of the code is needed.

7. Next steps

We will communicate the workshop findings with a broad audience in the coming months. This workshop report will be circulated to a range of stakeholders including modelers and funders.

Our findings will also be communicated through blog posts on the Software Sustainability Institute and NERC Environmental Data Services websites. We will convene discussions to outline a roadmap to take the recommendations of this report forward through the newly established special interest groups on models within NERC-EDS by the end of 2024. A webinar for all NERC EDS staff to report on the findings of this report is also being scheduled.

Follow-on surveys and discussions based on the recommendations from this workshop will be circulated to researchers, other end users, and stakeholders (e.g., policy makers) as we continue to understand different user needs.

Finally, please contact the authors if you have any comments or suggestions, or if you would like to contribute to any follow-on activities.

8. Acknowledgments

We would like to thank all the participants for their valuable insight. Thanks to NERC Environmental Data Service for its interest on this topic and the Software Sustainability Institute for funding this event via my fellowship award. Special thanks to Mary Preston for administrative support during the event and input from colleagues in the UKCEH Environmental Data Science group and Data Stewardship team.

9. Appendix 1: Agenda

Date and Time: 10am - 4pm, 21st November (Tuesday)

Venue: UKCEH Lancaster, Kestrel Rooms

Workshop planning: Michael Tso, John Watkins, Matthew Nichols, Rick Stuart, Philip Trembath, Kelly Widdicks, Kate Harrison, Rafael Barbedo, and *Reuben Margerison*, UKCEH and EIDC

Time	Item
9:30 -10:00	Arrivals and coffee
10:00-10:15	Welcome / logistics/ Opening remarks (Michael Tso)
10:15-10:45	Keynote 1: Prof Chris Jewell (Math and Stats, Lancaster University). The Generalised Epidemic Modelling (GEM) project: Automating infectious disease analysis
10:45-11:15	Lightning talks/ demos of tools (3 minutes each, questions during coffee) Gordon Blair: something around FAIR/environmental models Ed Rowe: sharing good practice in code and model development Rich Ellis: LTLS-FE code and data Alejandro Coca-Castro: Fostering open environmental science: The role of FAIR computational notebooks Tom Russell: Water Resources Modelling research and collaboration with regulators Charlotte Pascoe: Documentation for climate model simulations Ana Couto: Developing environmental impact assessment support tools to be used during long time periods. Kit Macleod: The changing nature and use of environmental model code
11:15-11:40	Networking and coffee
11:40-12:50	Group discussion 1: Capturing requirements of environmental code of long-term value Facilitators: Matt Nichols, Jennifer Roebuck, Sam Harrison, and Cansu Uluseker
12:50-13:30	Lunch: LEC Atrium

Time	Item
13:30-14:40	Group discussion 2: Opportunities and challenges for fostering environmental code of long-term value
14:35-14:55	Networking and coffee
14:55-15:45	Panel discussion What principles and actions can environmental modeller communities and NERC data centres take forward? A roadmap? Panellists: Tom Russell, Alejandro Coca-Castro James Bryne, and Gordon Blair
15:45-16:00	Summary of the day / next steps Closing remarks

10. Appendix 2: Resources and further reading

10.1 References

Barker, M., Chue Hong, N.P., Katz, D.S., Lamprecht, A.-L., Martinez-Ortiz, C., Psomopoulos, F., Harrow, J., Castro, L.J., Gruenpeter, M., Martinez, P.A., Honeyman, T., 2022. Introducing the FAIR Principles for research software. *Sci. Data* 9, 622.

<https://doi.org/10.1038/s41597-022-01710-x>

Garijo, D., Ménager, H., Hwang, L., Trisovic, A., Hucka, M., Morrell, T., & Allen, A. (2022). Nine best practices for research software registries and repositories. *PeerJ Computer Science*, 8, e1023. <https://doi.org/10.7717/peerj-cs.1023>

Siddorn, J. ; Blair, G. ; Boot, D.; Buck, J.; Kingdon, A. ; Kloker, A.; Kokkinaki, A.; Moncoiffe, G.; Blyth, E. ; Fry, M. ; Heaven, R.; Lewis, E. ; Marchant, B.; Napier, B. ; Pascoe, C.; Passmore, J.; Pepler, S.; Townsend, P.; Watkins, J. . 2022 An Information Management Framework for Environmental Digital Twins (IMFe). Southampton, NOC, 23pp.

<https://doi.org/10.5281/zenodo.7004350>

Wilson, G., Aruliah, D. A., Brown, C. T., Chue Hong, N. P., Davis, M., Guy, R. T., Haddock, S. H. D., Huff, K. D., Mitchell, I. M., Plumbley, M. D., Waugh, B., White, E. P., & Wilson, P. (2014). Best Practices for Scientific Computing. *PLoS Biology*, 12(1), e1001745.

<https://doi.org/10.1371/journal.pbio.1001745>

10.2 Video

- [Carole Goble on reproducible research --YouTube](#)
- [Supporting large scale modelling: guidance](#)
- [The importance of reproducibility in environmental science](#)
- <https://ncas.ac.uk/study-with-us/introduction-to-scientific-computing/>

10.3 Tools

- [Research Object Hub \(RoHUB\)](#): A EU-funded research object management platform that supporting the preservation and lifecycle management of scientific investigations, research campaigns and operational processes. It allows users to deposit and discover research objects and their linkages.

- [The Environmental Data Science book](#) includes many examples to apply data science methods to environmental data in Jupyter notebooks. All examples are issued with DOI and linked to RoHub.
- [FAIR-IMPACT](#) metrics for software includes a set of domain-agnostic metrics for software assessment and an automated tool for FAIR software assessment.

10.4 Case studies and articles

- [RDA tribute to Peter Fox \(standards creation and adoption in Earth Informatics\)](#)
- [Making UK weather and air quality data available to a diverse research community](#)
- [Using Open Source Software To Build Networks And Create Impact In Environmental Science](#)
- [SSI blog: Maintaining your legacy – tips for making legacy code sustainable](#)
- [SSI blog: Advocating for Recognition of Digital Output as Scholarly Artefacts](#)

10.5 Communities

- [ELIXIR-UK](#)
- [PANGEO](#)
- [Global Research Alliance Modelling Platform](#)
- <https://code.nasa.gov/>
- [CoMSES network](#): A growing collection of resources for computational model-based science
- [Open Geospatial Consortium](#) (OGC)
- [DataCite](#)
- [Earth Science Information Partners](#) (ESIP)
- [Zenodo](#): a general-purpose open repository developed under the European OpenAIRE program and operated by CERN
- [Australian Research Data Commons](#) (ARDC)
- [Making Models FAIR](#): A community initiative for making 100+ highly cited models findable, accessible, interoperable, and reusable (FAIR).
- [FAIRsharing.org](#): A curated, informative and educational resource on data and metadata standards, inter-related to databases and data policies.

Contact

enquiries@ceh.ac.uk

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

ceh.ac.uk

Bangor
UK Centre for Ecology & Hydrology
Environment Centre Wales
Deiniol Road
Bangor
Gwynedd
LL57 2UW
+44 (0)1248 374500

Edinburgh
UK Centre for Ecology & Hydrology
Bush Estate
Penicuik
Midlothian
EH26 0QB
+44 (0)131 4454343

Lancaster
UK Centre for Ecology & Hydrology
Lancaster Environment Centre
Library Avenue
Bailrigg
Lancaster
LA1 4AP
+44 (0)1524 595800

Wallingford (Headquarters)
UK Centre for Ecology & Hydrology
Maclean Building
Benson Lane
Crowmarsh Gifford
Wallingford
Oxfordshire
OX10 8BB
+44 (0)1491 838800



UK Centre for
Ecology & Hydrology