From Open Data to Reproducible Processes: FAIR Workflows

Approximate duration: ~40 min. The text is intended to be read aloud together with the presentation; interactions and demonstrations are included in the timing.

Key idea
Reproducibility is a property of the process, not of an individual file. A DOI for a result, without a description of how that result was obtained, is not enough to verify or repeat the study properly. The FAIR approach therefore extends from data to software, the execution environment, workflows, and provenance.

After this lecture, participants will be able to

- distinguish between open, FAIR, and reproducible;
- explain the difference between specification, run, and provenance;
- distinguish executable from descriptive workflows;
- identify the practical value of FAIR practices for their own research group;
- explain why transitions between stages are particularly vulnerable points.

Updated as of 2 November 2026
The National Open Science Plan, approved by Cabinet of Ministers Order No. 892-r of 08.10.2022, remains in force. Open-science practices and FAIR data are explicitly reflected in materials for 2026 calls of the National Research Foundation of Ukraine, but this should not be presented as a universal DMP requirement for every call. Horizon Europe requires a Data Management Plan and its updating during the project. On 8 October 2026, the EOSC Federation moved into production: 14 nodes became production-grade.

1. Motivation: “Your PhD student has left. Can you reproduce the calculation within a week?” [0–5 min]

On screen
One question + four cards: input/output; software; scripts/logs; decisions.

Let us begin not with the term FAIR, but with a very ordinary situation. A PhD student in your group has spent a year running calculations, producing tables and figures, training a model, and preparing a paper. The paper is published. A year later, the student moves to another institution. A new reviewer asks you to verify one point on a plot or repeat a calculation with a slightly different parameter. My question is very practical: could another person in your group reproduce the required part of the work within a week?

In most laboratories, the answer is not exactly “no”. The data are often somewhere. The file system contains input, output, scripts, results, and the PDF of the paper. But someone has to find the correct directory, identify which version of the code was used, distinguish a production run from a test run,

determine which of three scripts was the final one, and remember why two configurations were excluded. In other words, the files do not necessarily disappear. What disappears is the structure of knowledge about the process.

Think of this as a chain of evidence. Explaining a final result requires more than one file: input, parameters, software version, execution environment, intermediate artefacts, decision points, and the relationships among them. If even one critical link is missing, the result becomes difficult to verify.

This is why reproducibility in computational research is a property of the process rather than of an individual output. A DOI is extremely useful: it provides persistent identity and helps users find an object. But a DOI does not remember which executable was run, which environment was active, or why the researcher selected a particular checkpoint.

This will be the central thread of the entire course: how to move from “I have files” to “I have a research object in which it is clear what was done, what actually happened, and how one result gave rise to the next”.

Audience interaction

Run an anonymous poll or ask for a show of hands: who routinely preserves 1) input/output; 2) exact software versions; 3) the execution environment; 4) scripts/logs; 5) decisions and selection history? Do not grade the answers. Say that the group will return to this list in Lecture 4.

Transition to the next block

Now ask a different question: if this organisation requires additional effort, what does the researcher gain in return?

2. What does this give the researcher? FAIR as a way to reduce one’s own risks [5–10 min]

On screen

Four blocks: reproduce/hand over; resilience; grants; citation.

FAIR often sounds like an external requirement. A repository asks for metadata, a project asks for a DMP, and a journal asks for a data availability statement. If we look at FAIR only in this way, it is easy to perceive it as extra bureaucracy. I suggest a different framing: FAIR reduces the researcher’s own operational risk.

The first benefit is the ability to reproduce and hand over work. In a well-organised project, a new team member does not begin by asking, “Where is the final version?” They have a README, versioned code, a manifest, links between stages, and a clearly described environment. This does not guarantee that a complex calculation can be repeated immediately, but it substantially reduces reconstruction time.

The second benefit is resilience. For Ukrainian research, this is not an abstract issue. Servers, administrative teams, and hosting platforms can change, and long interruptions in access can occur. When critical artefacts have checksums, persistent identifiers, and independent copies, losing one workspace does not mean losing the group’s scientific memory.

The third benefit is the grant environment. In Horizon Europe, a Data Management Plan is part of data management and must be updated during the project. Open-science practices and FAIR data are already explicitly present in materials for 2026 calls of the National Research Foundation of Ukraine. We should not overstate this: the specific requirements of the Foundation must be checked for each individual call. But the general direction is clear - data stewardship is moving from “good advice” to a normal part of project management.

The fourth benefit is citation and collaboration. A dataset with a DOI, correct authorship, a licence, and a link to the article can be a research output in its own right. The same is increasingly true for a software release and a workflow. These create additional points of visibility for researchers and facilitate participation in international consortia.

So the argument that “FAIR is only needed by the funder” is weak. The research group itself receives the greatest day-to-day benefit: less dependence on memory, easier handover of work, less unnecessary repetition, and a stronger evidence base.

Transition to the next block

After the motivation, we can define briefly what FAIR means and why FAIR is not synonymous with openness.

3. FAIR in brief: Findable, Accessible, Interoperable, Reusable [10–15 min]

On screen

The four letters F/A/I/R; at the bottom: “Restricted data can still be FAIR”.

FAIR stands for Findable, Accessible, Interoperable, and Reusable. The principles were formulated by Wilkinson and co-authors in 2016. They were deliberately written to work for both people and machines.

Findable: an object has a persistent identifier and sufficient metadata, and the metadata are indexed or at least available for discovery. For a computational package, this means that not only the article but also data, software, and workflows can have their own identities.

Accessible: the user understands how to retrieve the metadata and content, via which protocol, and whether authorisation is required. This does not mean the content must always be public. Restricted data can still be FAIR if the metadata, access conditions, and request procedure are clear.

Interoperable: data and metadata use formats, vocabularies, identifiers, and relationships that other systems can interpret. This is where controlled vocabularies, QUDT units, PROV-O relations, DataCite identifiers, and domain ontologies become relevant.

Reusable: there is a licence or terms of use, provenance, context, and enough description for another researcher not merely to download a file, but to use it correctly. For an ML model, for example, this means knowing the training data, model version, validity domain, and limitations.

Why is it important to say “FAIR is not the same as open”? Otherwise, people working with sensitive or security-related data may immediately conclude that FAIR does not apply to them. In reality, FAIR concerns manageability, machine actionability, and conditions for reuse. Access may be restricted, but metadata and access rules should be clear.

Transition to the next block
FAIR began with data. But computational results also depend on software and workflows. The object of FAIR description therefore expands.

4. From FAIR data to FAIR software and FAIR workflows [15–22 min]

On screen
Diagram: FAIR DATA (2016) → FAIR SOFTWARE → FAIR WORKFLOWS (2020–2025).

If we preserve a dataset but not the software, the result may still be impossible to repeat. Code changes, libraries change defaults, and APIs disappear. This led to the FAIR for Research Software movement - FAIR4RS. The principle is straightforward: software also needs identity, version, authorship, licence, dependencies, and metadata.

The next step is the workflow. In 2020, Goble and co-authors proposed treating a computational workflow as a FAIR digital object. In 2025, Wilkinson and co-authors considered the application of the FAIR principles to workflows more systematically. This matters because the workflow is what connects data, software, and process.

At this point we must distinguish specification, run, and provenance. A specification describes what is supposed to be executed: steps, dependencies, declared inputs and outputs, and tools. A run is a concrete execution: exact files, exact parameter values, actual software versions, time, and compute allocation.

Provenance is the history of what actually happened and how generated entities are related to used entities and agents.

This corresponds to prospective and retrospective views. The prospective view answers “How is the process designed?” The retrospective view answers “What happened in this particular run?” In many laboratories, only the prospective description exists in a README or script, while the retrospective history is scattered across logs and directories.

For reproducibility, the two views need to be connected. A workflow specification may contain a training step, while provenance shows that the first training run failed, the second completed and generated checkpoint-17, and the third was a test. Without provenance, the statement “the workflow was run” does not identify which artefact became the scientific result.

The important conclusion is that a research object is not only a result, but also the history of how it was obtained. That history does not need to be stored in one file, but relationships among its parts need to be explicit.

Transition to the next block
Now we will look at two practical ways to represent a workflow: executable and descriptive.

5. Executable workflow and descriptive workflow [22–27 min]

On screen

Two columns: executable / descriptive. Examples: CWL, Snakemake, Galaxy, AiiDA versus shell + manual steps + heterogeneous tools.

An executable workflow is formalised to the point that a workflow engine can execute its steps again. CWL provides a declarative workflow description; Snakemake and Nextflow work well with file and task dependencies; Galaxy provides a GUI and History; AiiDA combines execution with a provenance graph.

This is a strong option, but it requires the process to be organised through a specific engine or language. That is often reasonable for a new project. For already-executed research or a heterogeneous laboratory workflow, the reality may be different.

For example, DFT may have been launched with shell scripts through Slurm, selection performed in a separate Python notebook, training launched by another command on a GPU node, model selection done manually, and MD run with another package. There is no single executable master workflow. But that does not mean the process cannot be described in a FAIR way.

A descriptive workflow is a structured record of a heterogeneous process. It may not automatically re-run the process, but it can capture stages, dependencies, inputs and outputs, provenance, and decision points. For many laboratories this is a realistic first step and a good foundation for later automation.

The aim of MATSCI-NASU is therefore not to force everyone immediately onto one engine. The aim is to preserve process context regardless of which tools were used to execute the work.

Parallel analogy for non-materials scientists

Any field can have a mixed pipeline: instrument export → Excel/manual cleaning → Python model → web service. FAIR description can begin before the workflow is fully automated.

Transition to the next block

When many tools are involved, the useful question is not “Which one is best?” but “What function does each one perform?”

6. International and European practices: an ecosystem, not a single platform [27–32 min]

On screen

Six cards: WorkflowHub/Dockstore; RO-Crate/WRROC; Dataverse/Zenodo; NOMAD/FAIRmat; AiiDA/Materials Cloud; OPTIMADE.

WorkflowHub and Dockstore provide a registry and discovery layer: they help users find workflows and describe their versions. RO-Crate provides a packaging model for research artefacts, while Workflow Run RO-Crate specialises in execution records. Dataverse and Zenodo provide repository and preservation functions.

In materials science, FAIRmat/NOMAD provide a domain infrastructure with parsers, normalisation, and search across computational materials data. AiiDA provides automatic provenance for

calculations executed through AiiDA and is closely connected with Materials Cloud. OPTIMADE provides an interoperable API for materials databases.

These systems are not interchangeable. A repository is not a workflow engine. A workflow engine is not necessarily a long-term publication repository. A domain normaliser does not always preserve manual decisions between heterogeneous stages. We therefore need architectural thinking: execution, provenance, package, repository, and discovery are different layers.

For this course, it is enough to remember one rule: you do not need to use everything. You need to know what problem your workflow has. If the issue is reproducible execution, look at a workflow engine. If the issue is preservation, look at a repository. If the issue is domain normalisation, look at NOMAD. If the issue is cross-platform packaging, look at RO-Crate/WRROC.

In Lecture 2 we will return to this point and show why MATSCI-NASU is positioned as a domain package profile rather than a new workflow manager.

Transition to the next block

Now for the most important bridge to the next lecture: even when stages are well described, information loss often occurs on the arrows between them.

7. The most vulnerable places are transitions between stages [32–36 min]

On screen

DFT outputs → training set → MLIP model → MD trajectory; transitions highlighted.

In a single-step calculation, the relationship is often obvious: input → code → output. In DFT → MLIP → MD, the data change role at every boundary. DFT outputs are not yet a training dataset. Labels must be extracted, failed runs may need to be filtered, structures may need to be deduplicated, a format must be created, and a split must be defined.

After MLIP training, there is not automatically one unique model either. There may be several checkpoints, metrics, and criteria. Someone selects a model, and sometimes compiles or deploys it for another runtime. MD then uses a specific artefact. If we record only “MLIP model”, lineage remains ambiguous.

Human decisions are especially important. A researcher may reject configurations because an SCF calculation failed to converge, inspect a trajectory visually, or select a checkpoint based on physical stability rather than the lowest loss. Such decisions often never appear in logs.

The thesis of Lecture 2 will therefore be: a transition is not a decorative arrow. If it changes data, selection, or interpretation, it should be described as a separate object or activity.

Parallel analogy for non-materials scientists

Data collection → cleaning/labeling → training → prediction. The least reproducible part is often cleaning: which rows were removed, who identified an outlier, and which version of the script was used.

Transition to the next block

We will finish Lecture 1 with the current Ukrainian and EOSC context and three points that participants

should retain.

8. Ukrainian and EOSC context. Summary [36–40 min]

On screen

Current fact: 08.10.2026 EOSC Federation → production; 14 production nodes. Three concluding messages.

In Ukraine, the 2022 National Open Science Plan remains in force. In 2026, open practices and FAIR are becoming increasingly explicit in grant and institutional processes. DataverseUA provides a national repository layer, while competence centres and training materials should help turn policy into everyday practice.

At the European level, the situation has also changed. On 8 October 2026, the EOSC Federation moved into production: the EOSC EU Node and 13 pioneering nodes became production-grade. This does not mean the Federation is complete - other nodes continue readiness work, service onboarding, and development of federation capabilities. But teaching materials should now describe EOSC as an operational federation that continues to evolve, rather than only as a future concept.

What should remain after Lecture 1? First: FAIR ≠ open, and open ≠ reproducible. Second: a computational research object includes data, software, the environment, workflow, and provenance. Third: the greatest information loss occurs at transitions and in human decisions. Lecture 2 therefore begins with the model “stages + transitions + plan/fact/inventory”.

Return mentally to the opening poll. If you preserve only input and output, that is already better than preserving nothing. But after this lecture, add exact software and environment, processing scripts, transition artefacts, and decision history. This is the first practical step from files to a FAIR research process.

Audience interaction

Ask participants to name one thing they would add to their usual way of preserving a project after this lecture.

Sources and materials

1. Wilkinson M. D. et al. (2016). The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data 3, 160018. https://doi.org/10.1038/sdata.2016.18

2. Goble C. et al. (2020). FAIR computational workflows. Data Intelligence 2(1–2), 108–121. https://doi.org/10.1162/dint_a_00033

3. Wilkinson S. R. et al. (2025). Applying the FAIR Principles to computational workflows. Scientific Data 12, 328. https://doi.org/10.1038/s41597-025-04451-9

4. Leipzig J. et al. (2021). The role of metadata in reproducible computational research. Patterns 2(9), 100322. https://doi.org/10.1016/j.patter.2021.100322

5. European Commission. Horizon Europe Data Management Plan template and Model Grant Agreement guidance.

6. National Research Foundation of Ukraine. Conditions of 2026 calls; Science Europe–NRFU materials on open science and FAIR data.

7. Cabinet of Ministers of Ukraine. Order No. 892-r of 08.10.2022 “On approval of the National Open Science Plan”.

8. EOSC Association. EOSC Federation moves into production, 8 October 2026.