FAIR hackathon 2021

Session 3: Workflow & Tools
FAIRification

Carole Goble
Stian Soiland-Reyes
Finn Bacall
Stuart Owen
Douglas Lowe
Nick Juty
Simone Leo

EOSC-Life WP3 FAIRification workshop
2021-12-16
Schedule

- Overview of workflows and workflow management software
- Brief Mentimeter survey of participants
- Making my Workflows FAIR
- Incorporating my tool and making it FAIR Workflow ready
- Final discussion

Please help take collaborative notes!
A data and method commons

A portable environment of interoperable tools

Workflows are an entry point to the tools and datasets
functions for production quality FAIR data processing
access to secure data processing

RIs publish data, methods & services for management, storage and reuse

Figure Credit: Romain Dallet
Galaxy Genome Annotation (GGA) environment in the cloud
Mentimeter questions for the participants

Do you already use workflows?
   If yes, which systems?
   If no, which systems do you intend to use?

Do you have tools that need to be wrapped to run in workflows?
   If so, are they command line, web services, containers?

Will you be using sensitive data?
Will you need to operate in a Trusted Research Environment?

What kind of provenance do you need to collect?
   • data lineage (traceability of data)
   • workflow execution (recomputability of data)

Is computational portability important?

Do you have test cases set up for your tools / workflows?
What is a Workflow?

Why use a Workflow Management System

What kind of systems are available for what kind of problems?
Workflows are an entry point to Computational Science and a way to run & share multi-step methods

- Multi-step (semi)-automated data processing pipelines, simulation studies and analytics
- Access to (distributed) data, tools; Scalable use of computational resources
- Simplified tool chains, auto-documentation, shielded from technical heroics
- Shared, validated and comparative know-how
Why are Workflow Systems Useful

Abstraction & Composition
- Using the best codes written by 3rd parties
- Handle heterogeneity
- Shield complexity & incompatibility
- Sharable reusable, re-mixable methods

Automation
- Repetitive reproducible pipelines
- Simulation sweeps
- Manage data and control flow
- Optimised monitoring & recovery
- Automated deployment

Sharing & Adaptability
- Shared method, publishable know-how
- BYOD / parameters
- Different implementations
- Changes in execution infrastructure

Scalability & Infrastructure Access
- Accessing infrastructures, datasets and tools
- Optimised computation and data handling
- Parallelisation
- Secure sensitive data access & management
- Interoperating datasets & permission handling

Reporting & Accreditation
- Provenance logging & data lineage
- Auto-documentation
- Result comparison

Portability
- Dependency handling
- Containerisation & packaging
- Moving between on premise & cloud
What benefits to me does a Workflow Management System offer?

Heavy lifting
• The WfMS handles the tricky infrastructure bits and dependencies

Documented method
• Process, debug and record
• Publishable results

Collective working
• Building up a pool of know-how
• Variant-based development - reuse, remix, repurpose

There is an upfront cost for downstream benefits

Benefits are best when a community buys in and workflows are supported
Workflow System Landscape - what do you use/need?
Different systems have their different strengths

Interactive Electronic Research Notebooks

RStudio

Scripting environments

Inter-twiningling Mix and Matching

Workflow Management Systems & execution platforms

Interactive & exploratory analysis

Production, automated, workflow-integrated software

https://s.apache.org/existing-workflow-systems
Frameworks to web based analysis platforms, hybrid cloud deployment

Workflows are rules:
Graph of jobs for automatic parallelisation, DIY package & containerisation installation, auto-documentation
Open framework, DSL and execution engine using file systems and data outside

Self-contained online VRE. Users build and reuse workflows around publicly available or user-uploaded data and pre-wrapped, pre-installed tools.

Communities tend to cluster round a few systems.
Take up of a WfMS typically depends on the “plugged-in” availability of data type specific codes, skills level of the workflow developers, and popularity.
Workflow stakeholders - who are you?
Are your workflows one-offs or production?

- **Sys Admin**
  - infrastructure & services

- **Tool Developer**
  - tools to be wrapped & maintained

- **Workflow Developer & Custodian**
  - workflows developed, tested, run & maintained

- **Computational User**
  - find and understand workflows, with explanations to use properly and safely.

- **Workflow User**

- **Workflow Application User**

Use reach
Questions

What workflow system do you use or need?

What kind of workflow user are you?
Now you know what a workflow is!

Mentimeter information gathering

https://www.menti.com/87ouhg158g or

Go to www.menti.com and use the code 4885 3343
Making my Workflows FAIR
FAIR Principles applied to Workflows

https://doi.org/10.1162/dint_a_00033
Workflows should produce FAIR Data Products

- Metadata generated and exposed for data products?
- Workflow consumes, propagates, generates
- FAIR identifiers for data?
- license data outputs?
- community data formats?
- usage restrictions on the reference data needed?
- parameters validated?
- How does the workflow access FAIR data?
- fully track and report data provenance through the workflow?
Workflow FAIR Data by Design

Assisted by WfMS, Challenge of diverse API & AAI landscape, formats and packaging

Design for FAIR Data and Reuse

- Workflow consumes, propagates, generates FAIR identifiers for data?
- How does the workflow access FAIR data?
- Usage restrictions on the reference data needed?
- Community data formats?
- License data outputs?
- Parameters validated?
- Fully track and report data provenance through the workflow?

Canonical workflows

Metadata generated for data products

Best Practice Golden Examples

- Reviewing Curation Certification Governance

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 824087
Things to consider when making workflows to make sure the data their produce adheres to the FAIR principles

Is the input data publicly accessible, and if so where?

Will the output data be publicly accessible, and if so where?
Will the data be licensed?

Is metadata generated for the data products?
• Do you have to collect the intermediate data products too?

Is data traceable?
• combination of DIY provenance in the workflow + the WfMS support
Workflows should be FAIR objects and FAIR software

**FAIR for Research Software (FAIR4RS) Working Group**

Katz, et al PATTERNS 2, 2021

Lamprecht et al., 2020

Goble et al, 2020

register workflows with assigned PID + metadata in a searchable resource.

metadata & workflow retrievable by PID using a standardized communication protocol; metadata retrieval even if the workflow is no longer available.

workflows should read, write or exchange data using domain-relevant community standards

The workflow is **usable** (it can be executed) & **reusable** (it can be understood, modified, built upon, or incorporated into other workflows).

(Meta)data, workflows and tools are associated with **detailed provenance** – data lineage, workflow lineage & workflow logs
We need Workflow Services to be FAIR and we need FAIR Enabling Services

Honour legacy & diversity by supporting native repositories & platforms

On-board systems and services:

- lifting and sharing common ids & metadata
- adopting common APIs

Enable FAIR workflow design
Workflows are Findable & Accessible

Registered Digital Objects of Scholarship
registered with PID + metadata in a searchable resource
• Versioned, DOI/PID assignment
• Collections, workflow libraries
• Reviewed and Published

Access, Licence & Execute
accessed from the PID
• Download
• Execution API (GA4GH)
• Metadata accessible even if workflow un-runnable

Native systems & repositories
WORKFLOW SYSTEMS
- Galaxy PROJECT
- KNIME
- SCIPION
- nextflow

INFRASTRUCTURE
- PULSAR PROJECT
- Kubernetes

WF REPOS
- Github
- nf-core

REGISTRIES
BioContainers
elair
Bio.tools
WorkflowHub
Publishing Services Repositories
Knowledge Graphs

Research Infrastructure
registries
BioContainers
elair
Bio.tools
WorkflowHub
Publishing Services Repositories
Knowledge Graphs

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 824087
Workflows are Interoperable, Reusable and Usable

Portability

Workflow and tool collections & libraries to share, reuse and remix.

Workflow benchmarking, monitoring, testing

https://workflowhub.org

https://lifemonitor.eu/
https://openebench.bsc.es/dashboard
The FAIR principles are all about human and machine actionable metadata circulating between the services and exported outside.

System neutral workflow description
Input and output types
Metadata markup about the workflow, tools & parameters
FAIR Digital Workflow Object (FDO) packaging and linking a workflow with all its bits, provenance and companions for exchange, archiving, publishing and citation

We hide most of this!
What can you do to be FAIR? We can help each other

- Register/Publish
  - Cite & credit makers

- Skill the Team with Best Practice

- Document for Strangers
  - Use Standards
  - Use IDs

- Use WfMSs and tools that are FAIR enabling

- A Management Plan Checklist

- Get Help
  - Professionalisation
  - Pre and post hoc
Registering a Workflow in WorkflowHub Using benchmarking and testing tools
WorkflowHub
https://workflowhub.eu

Findable & Accessible
register workflows with assigned PID + metadata in a searchable resource

DOIs on workflows

Publishable and citable objects

curated collections

Faceted Search
Rich metadata
Findable and Accessible

Curated Collections
Associated objects
Teams, Credit

metadata & workflow retrievable by PID using a standardized communication protocol

register workflows with assigned PID + metadata in a searchable resource
Organise workflows
Teams and People
Workflow lifecycle support

Curation

Manual (e.g. collections)
Automated (e.g. GitHub)
By stealth (e.g. from WfMS)
By services (e.g. monitoring & testing)
Credit for curators?

The Journal of Open Source Software
What can I register in WorkflowHub?

• Workflows!
  • Including scripts, Jupyter notebooks etc.
• Presentations
• Publications
  • Import with DOI or pubmed ID
• Data sets
  • Register example/test data and associate with a workflow
• Documents
  • Curated collections of all the above
  • Gather related workflows etc. together

https://workflowhub.eu/collections/3
How can I register a workflow / notebook / script in WorkflowHub?

• **Multiple ways to register**
  • **Upload** files from your computer
  • **Link** to existing files on the web
  • **Import** **git** repositories

• **Flexible type system**
  • Numerous **workflow types** supported
    • Metadata extraction or Abstract CWL
    • Diagram generation or upload
  • Users can add additional types if needed
  • Scripts and Notebooks welcome!
WorkflowHub & GitHub(/lab) Coming Soon

- “Pull” workflows directly from Git repositories
- “Push” workflows automatically through GitHub actions etc.
- Automatic parsing of metadata from repositories:
  - Workflow files
  - LICENSE
  - citation.cff
  - ro-crate-metadata.json
What metadata do I need to register a workflow?

Minimal required metadata, extend incrementally

Workflow Type: New workflow type
PyCOMPSs

Title:
Protein MD Setup HPC tutorial using BioExcel Building Blocks (biobb) in PyCOMPSs

Source:
https://github.com/bioexcel/biobb_hpc_workflows
If this workflow came from an external repository (i.e. GitHub), you can include its original URL here.

Maturity:
Stable
This field is used to indicate to users what level of stability they can expect from the workflow.

License:
Apache Software License 2.0
For more information on this license, please visit https://opensource.org/licenses/Apache-2.0

Description:
This PyCOMPSs workflow tutorial aims to illustrate the process of setting up a simulation system containing a protein, step by step, using the BioExcel Building Blocks library (biobb) in PyCOMPSs for execution on HPC. Three variants of the MD Setup workflows are included, supporting a list of structures, a list of mutations, or a cumulative set of mutations.

Creators:
Adam Hospital (BioBB Building Blocks) ✗
Pau Andrio (BioBB Building Blocks) ✗

If anyone not registered with WorkflowHub assisted creating this Workflow, you can specify them below.

Genis Bayarrí

For more information, please visit https://bioschemas.org
What *extra* metadata could I add for a workflow?

**Discussion Channels**

You can add one or more URL's to point to discussion channels related to this Workflow:

- https://ask.bioexcel.eu/c/bioexcel-building
- http://mmb.irbbarcelona.org/biobb/worklist
- https://biobb-wf-md-setup.readthedocs.io
- https://github.com/bioexcel/biobb_hpc_wr

**Attributions**

If this Workflow is based on any existing Workflows, please list them below.

So far you have specified the following attributions:

*Workflow: Protein MD Setup tutorial using BioExcel Building Blocks (biobb) (Douglas Lowe)*

Please type titles of Workflows into the box below - suggestions will be displayed as you type. Select resources that you want to attribute to.

**Tags**

You are able to edit all tags, including both the tags you have added and tags added by other people. Any new tags you add will be added as your own tags. Tags you remove, even those added by other people, will be completely removed from this Workflow. Tags should be separated by a comma, and known tags will appear in the drop-down box as you type.

- molecular dynamics
- GROMACS
- BioBB

**Publications**

The following Publications are associated with this workflow:

- BioExcel Building Blocks, a software library for interoperable biomolecular simulation workflows

Select Publication...
Quality assessment, amplicon classification and functional prediction

Version 2

Workflow Type: Common Workflow Language

Workflow for quality assessment of paired reads and classification using NGTax 2.0 and functional annotation using picrust2. In addition, their respective subfolders for easier data management in a later stage. Steps:

- FastQC (read quality control) - NGTax 2.0 - Picrust 2 - Export module for ngtax

Inputs

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>forward_reads</td>
<td>forward sequence file locally</td>
<td>File</td>
</tr>
<tr>
<td></td>
<td>reverse_reads</td>
<td>reverse sequence file locally</td>
<td>File</td>
</tr>
<tr>
<td></td>
<td>forward_primer</td>
<td>Forward primer used</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>reverse_primer</td>
<td>Reverse primer used</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>reference_db</td>
<td>Reference database used in PASTA format</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>rev_read_len</td>
<td>Reverse read length</td>
<td>int</td>
</tr>
<tr>
<td></td>
<td>ref_read_len</td>
<td>Reverse read length</td>
<td>int</td>
</tr>
<tr>
<td></td>
<td>sample</td>
<td>Sample name</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>fragment</td>
<td>Subfragment name</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>primersRemoved</td>
<td>Primers are removed</td>
<td>boolean</td>
</tr>
<tr>
<td></td>
<td>threads</td>
<td>number of threads</td>
<td>int?</td>
</tr>
<tr>
<td></td>
<td>metadata</td>
<td>Metadata file</td>
<td>File?</td>
</tr>
</tbody>
</table>
What WorkflowHub features can be useful for my project and workflows?

• Make your workflow **discoverable**
• **Catalogue** of workflows from a project/group/organization
• “Sharing” **permissions** let you decide who can access your workflow
  • Share first with collaborators, publish when ready
• Indicate your workflow **status** as a “Work in progress” or “Stable”
• Register new **versions** as your workflow evolves
• Generate a **DOI** for your workflow and make it citable
• **Relate** workflows to ongoing and published **research**
• RO-Crate and Bioschemas generated with **FAIR metadata**
Questions to ask yourself

Will you have some workflows ready to register during the hackathon?

Does anything prevent your workflow being published and made open?

Will multiple people be collaborating openly on the workflow?

Do you know the names and affiliations of the workflow authors?

Have you considered the license for your workflow?
https://www.software.ac.uk/resources/guides/choosing-open-source-licence
How can I test my workflows?

- **Functional testing**
  - Does the workflow run with no errors?
  - Does it produce the expected outputs?

- **Test cases**
  - Small, simple inputs → easy to make assertions on expected outputs
  - Tests should run relatively fast and not involve randomness (e.g., set seed)

- **Dedicated frameworks simplify setting up test cases**
  - Planemo (Galaxy and CWL)
  - Pytest-workflow (engine-agnostic)

- **CI services automate test execution and provide isolated environments**
  - GitHub Actions, Jenkins, Travis CI, ...

- The **LifeMonitor** team is working with **Galaxy IWC**
  - Integration of workflow best practices with WorkflowHub and LifeMonitor
  - Testbed for more general workflow maintenance support framework
Incorporating my tool and making it FAIR Workflow ready
10 Simple Rules for making a software tool workflow-ready

Paul Brack; Peter Crowther; Stian Soiland-Reyes; Stuart Owen; Douglas Lowe; Alan R Williams; Quentin Groom; Mathias Dillen; Frederik Coppens; Björn Grüning; Ignacio Eguinoa; Philip Ewels; Carole Goble

Workflows have become a core part of computational scientific analysis in recent years. Automated computational workflows multiply the power of researchers, potentially turning “hand-cranked” data processing by informaticians into robust factories for complex research output.

However, in order for a piece of software to be usable as a workflow-ready tool, it may require alteration from its likely origin as a standalone tool. Research software is often created in response to the need to answer a research question with the minimum expenditure of time and money in resource-constrained projects. The level of quality might range from “it works on my computer” to mature and robust projects with support across multiple operating systems.

Despite significant increase in uptake of workflow tools, there is little specific guidance for writing software intended to slot in as a tool within a workflow, or on converting an existing standalone research-quality software tool into a reusable, composable, well-behaved citizen within a larger workflow.

In this paper we present 10 simple rules for how a software tool can be prepared for workflow use.

Accepted at PLOS Computational Biology (PCOMPBIOL-D-21-01704)

Rule 1: Make sure a workflow engine can talk to your software easily
Rule 2: Make your tool simple to install
Rule 3: Document your tool
Rule 4: Make your tool maintainable
Rule 5: Follow the principle of least surprise
Rule 6: Make your tool parallelizable
Rule 7: Make your workflow tool a good citizen
Rule 8: Make output reproducible
Rule 9: Carefully consider human interaction
Rule 10: A software tool should just do one thing

https://doi.org/10.5281/zenodo.5636487
WfMS and tool interaction

Rule 1: Make sure a workflow engine can talk to your software easily
Rule 2: Make your tool simple to install
Rule 8: Make output reproducible
Rule 9: Carefully consider human interaction

Tool configurable at runtime
Software and library dependencies should be explicit
Use a package manager

Does your software tool make it simple for researchers to include all of the data, methodology and software tooling to allow another researcher to recreate research findings?

Wrapping a tool for a workflow can end up hiding visualizations and interactions with users.
Human-in-the-loop sacrifices automation and reproducibility
User inputs must be reported as parameters or choices, so this is made accessible as provenance and potentially automatable later.

https://doi.org/10.5281/zenodo.5636487
WfMS composability

Rule 1: Make sure a workflow engine can talk to your software easily
Rule 5: Follow the principle of least surprise
Rule 10: A software tool should just do one thing

All input and output data is computer-addressable
Input and output file(s) or directory(ies) always be specified as arguments at runtime.

Inputs to and outputs clearly and explicitly named, tool’s behaviour match documentation

Use standard streams in an accepted manner

Where software performs several different discrete tasks, wrap the executable in several different wrappers and implement different endpoints as different workflow tools.

https://doi.org/10.5281/zenodo.5636487
Rule 6: Make your tool **parallelizable**
Rule 7: Make your workflow tool a **good citizen**

Do multiple copies of the running software *overwrite* one another’s runtime file structures?
Tool accesses *external APIs* within access limitations.

Design software likely to be run on *shared infrastructure* than on a single user’s computer. Reasonably *performant* and should not reserve more *system resources* than needed. *Disk usage* post-run and clear up any *temporary files*.

https://doi.org/10.5281/zenodo.5636487
Rule 3: Document your tool
Rule 4: Make your tool maintainable

Interface-level documentation
Provide code snippets

Source control software (e.g. git)
Version management
Source code archive, installation package
Unit and Integration tests

https://doi.org/10.5281/zenodo.5636487
Acknowledgements

Join the WorkflowHub Club

https://about.workflowhub.eu/community/

WorkflowHub https://workflowhub.eu/
EOSC-Life https://www.eosc-life.eu/
ELIXIR http://elixir-europe.org
RO-Crate https://www.researchobject.org/ro-crate/
Galaxy Europe https://galaxyproject.eu/
Bioschemas https://bioschemas.org
CWL https://www.commonwl.org/
WorkflowsRI https://workflowsri.org/
Dockstore https://dockstore.org/
LifeMonitor https://lifemonitor.eu/
BY-COVID https://by-covid.org/
Questions?