

ICFA Data Lifecycle, EVERSE and HSF

Mapping software quality criteria across communities

Caterina Doglioni (University of Manchester)

Input/slides from:

Stefan Roiser (CERN), Eduardo Rodrigues (Liverpool), Michael Sparks (UNIMAN), Graeme Stewart (CERN),
Fotis Psomopolous (CERTH)



Funded by
the European Union

29 | 04 | 2026.- HSF Seminar



You've just seen the ICFA lifecycle recommendations, now a refresher on **EVERSE**

Paving the way towards a
European **V**irtual Institute **e** for **R**esearch **S**oftware **E**xcellence

EVERSE aims to create a framework for research software and code excellence, collaboratively designed and championed by the research communities, in pursuit of building a European network of Research Software Quality and setting the foundations of a future Virtual Institute for Research Software Excellence

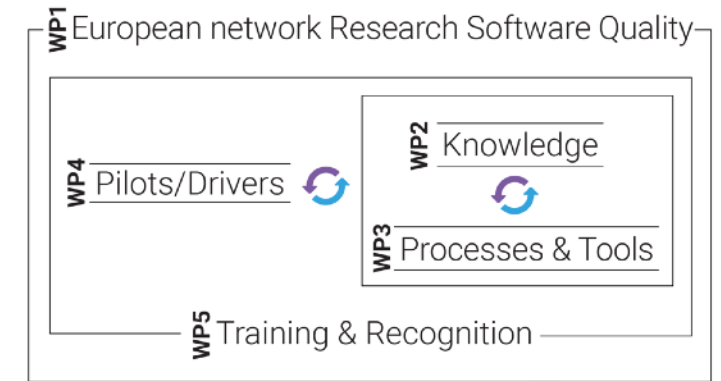
- ✓ ensure research software curation, quality, preservation and adoption of best practices, **by the Communities, for the Communities**, build on collaboration with the five **EOSC Science Clusters**
- ✓ adopt a **three-tier model for research software**, i.e., analysis code, prototype tools and research software infrastructure, which captures the varying complexity of research software and its development, and can be used as a basis for research software excellence
- ✓ **credit** and **recognition** for both developers and software are essential components of our strategy to promote sustainable software practices

Mar/2024 → Feb/2027 (36 months)

15 Beneficiaries, 1 Associated partner & 2 Affiliated entities

Coordinated by CERTH and BSC

How EVERSE works



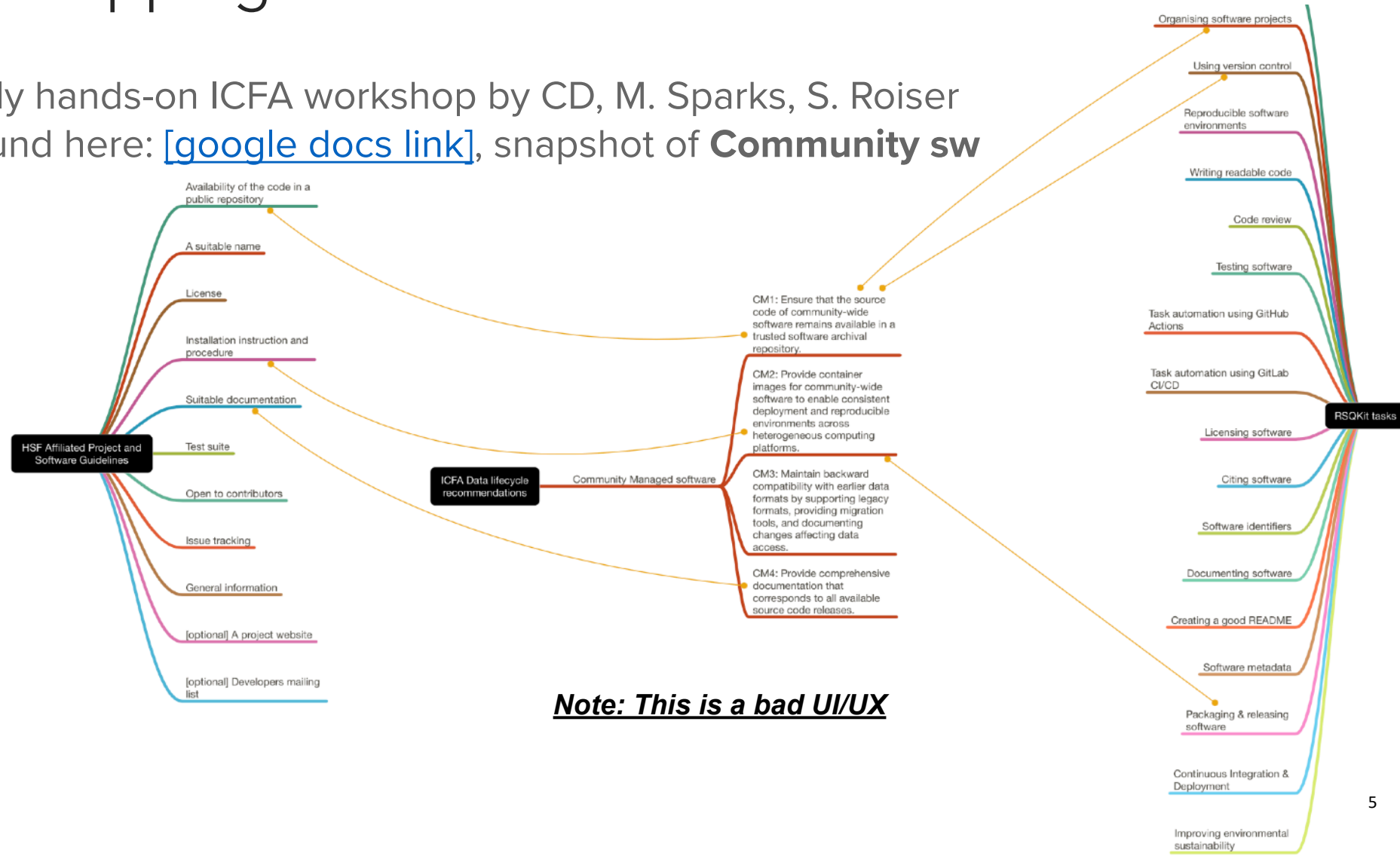
- Knowledge about software best practices is captured and distilled in **WP2: Community-led best practices for developing high-quality research software**
 - The **Research Software Quality Kit**, a.k.a. the **RSQkit** is the concrete output of commonly applicable practices → **this is our starting point for the mapping of best practices,** contains a **task driven** guide on **how** to improve research software quality
- Tools to help ensure software quality are identified and curated in **WP3: Tools and services for software quality and FAIRness**
 - Help integration into **software quality pipelines** and develop **quality indicators** that can be used by the communities
- Informing and testing these outputs are the **WP4: EVERSE Pilots and Drivers**

Commonalities/differences

Aspect/Goal	ICFA Role (Strategic "Why")	HSF Role (Project "How")	RSQKit Role (Developer "How-To")
Sustainability	Mandates long-term data and software preservation for future use.	Provides guidelines for community-driven projects that can outlive a single developer (licensing, contribution guides).	Provides tasks (testing, CI, documentation) that make software easier for others (or future-you) to maintain and build upon .
Reproducibility	States that analyses must be reproducible as a core scientific principle.	Requires version control and clear release procedures, which are foundational for reproducibility.	Gives specific tasks : "Tag your releases in Git," "Use a container for your environment," ensuring a reproducible setup can be built.
Quality	Implies high-quality software is needed for reliable scientific results.	Encourages practices like testing and CI that lead to community trust and higher quality contributions.	Provides a direct checklist to improve code quality: code coverage metrics, static analysis, unit tests.
Documentation	Requires documentation of software and analysis workflows for preservation and reuse.	States a project <i>must</i> have a README and <i>should</i> have user/developer documentation	Gives concrete tasks : "Write a good README.md," "Use a documentation generator like Sphinx or Doxygen."
Collaboration & Reuse	Promotes open data and open software for wider community benefit.	Sets the social and technical framework for open collaboration (Code of Conduct, license, contribution guidelines).	Encourages practices (packaging, clear APIs) that make it easier for others to actually use and integrate your software.

How does the "mapping" across these efforts look like?

- Work done during the July hands-on ICFA workshop by CD, M. Sparks, S. Roiser
- Full document can be found here: [\[google docs link\]](#), snapshot of **Community sw**



Note: This is a bad UI/UX

In practice: a test “joint” HSF/ICFA review (Madgraph)

- Exercise undertaken by CD / Stefan Roiser for Pythia and Madgraph [link]
 - Act as reviewers for projects that want to become affiliated, do the HSF review first
 - Then map the review outcomes to the ICFA data lifecycle recommendations
-
- LC1: Mark intellectual property ownership with a copyright statement in accordance with host laboratory and home institute rules.
 - LC2: Apply a license approved by the Open Source Initiative (OSI) to your software
 - <https://opensource.org/license/uoi-ncsa-php>, approved by the Open Source Initiative (OSI)
 - LC4: Provide methods to make software products citable.
 - Citation to paper describing the software clearly evidenced in README
 - LC5: Give priority to open-source software and tools with appropriate licences, and cite software appropriately in publications.
 - The Madgraph software has 16000+ citations
 - External libraries are embedded in the software repository (with licenses)
-
- CM1: Ensure that the source code of community-wide software remains available in a trusted software archival repository.
 - GitHub repository: <https://github.com/mg5amcnlo/mg5amcnlo>
 - Launchpad: <https://launchpad.net/mg5amcnlo>
 - CM2: Provide container images for community-wide software to enable consistent deployment and reproducible environments across heterogeneous computing platforms.
 - Containers exist (e.g. https://github.com/scaifin/MadGraph5_aMC-NLO, including for GPU) but not provided by maintainers
 - It makes sense that end users create these containers, since personalisation of models beyond the default included with the software is important for reproducibility
 - CM3: Maintain backward compatibility with earlier data formats by supporting legacy formats, providing migration tools, and documenting changes affecting data access.
 - Long term stable releases exist: <https://launchpad.net/mg5amcnlo/+announcement/54113>
 - CM4: Provide comprehensive documentation that corresponds to all available source code releases.
 - Documentation exists for all releases through README files

PR to be reviewed: HSF points to “why” (data lifecycle)

- On the [“HSF affiliated projects” page](#), add text & links to
 - ICFA data lifecycle [mission](#) and [mandate](#)
 - ICFA data lifecycle [arxiv recommendations](#) (which links to the [app](#))

HSF Affiliated Projects and Software

Introduction

The HEP Software Foundation is delighted to support and promote community software for high-energy physics. Here we describe how we can help projects gain greater visibility and enhance the positive impact of their software. ←

HSF Affiliated Projects and Software are community-driven and community-oriented “endeavours” or “products” of wide and recognised interest and applicability beyond a single collaboration or experiment.

HSF Affiliated Projects and Software are community projects or software packages with person-power, and possibly dedicated funding, which connect strongly to the HSF and align with *its goals*. They benefit from their inclusion in, or association with, the HSF through access to the community for wide visibility and easier promotion through the network. Affiliation makes it clear that the HSF has no direct control over the endeavours/products, hence no responsibility for evolution or maintenance.

Projects are hosted on their own public platform of choice (such as GitHub); they may use the [HSF GitHub repository](#) if desired and mutually agreed with the HSF Steering Group.

Text that is being added: Affiliated projects also align with several of the software-related recommendations from the ICFA (International Committee for Future Accelerators) data lifecycle panel [[mission](#)] [[mandate](#)] [[recommendations](#)][[app](#)].

- The HSF affiliated project could be mentioned on the data lifecycle panel website as well

Backup slides

Contact: contact@everse.software

EVERSE Network

<https://ec.europa.eu/eusurvey/runner/EVERSENetworkJoinIndividual>

Website:

<https://www.everse.software/>

BlueSky:

<https://bsky.app/profile/eosc-everse.bsky.social>

LinkedIn:

<https://www.linkedin.com/company/eosc-everse/>

FOSSTodon:

https://fosstodon.org/@eosc_everse



**Funded by
the European Union**

This project has received funding from the European Union's Horizon Europe Programme under GA 101129744 – EVERSE – HORIZON-INFRA-2023-EOSC-01-02



Why focus on software?

- Software is a critical part of the data lifecycle (and of the scientific process)
- From simulation / digital twins, to data acquisition and analysis of results software is vital to transform **data** into **knowledge!**
 - 92% of academics use software tools; 70% of researchers say their work would be impossible without software*
- The size of the software required to support recent experiments is growing
 - Our organisations / funders are still getting to grips with software recognition
- Consequently, **software quality really matters to us**

[*https://www.software.ac.uk/blog/its-impossible-conduct-research-without-software-say-7-out-10-uk-researchers](https://www.software.ac.uk/blog/its-impossible-conduct-research-without-software-say-7-out-10-uk-researchers)

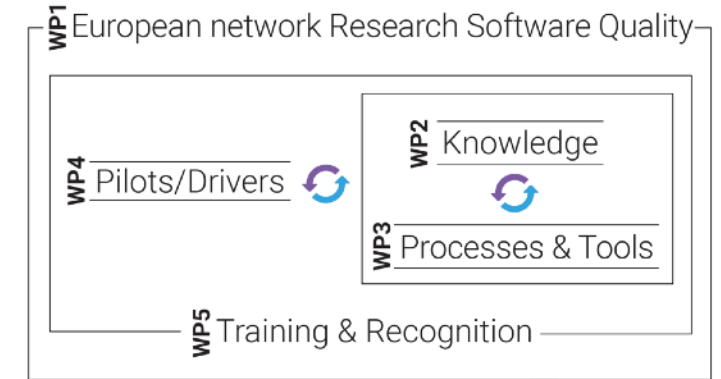
Connection between FAIR data and software

- We also strive to make our software FAIR and **sustainable**
 - *Findable; Accessible; Interoperable; Reusable* ([FAIR4RS paper](#))
 - Sustainable: “[software that] will continue to be available in the future, on new platforms, meeting new needs” (Dan Katz)

<https://icfa.hep.net/wp-content/uploads/ICFA.DataLifecycle.202509.pdf>

In accordance with the FAIR principles for scientific data management and stewardship (“Findable, Accessible, Interoperable, and Reusable”), the recommendations emphasize the significant long-term scientific value of experimental data and the importance of supplementary information and knowledge necessary for their understanding and (re)use. They also highlight the importance of analysis software and analysis workflow descriptions as integral parts of the research outcome.

How EVERSE works - 2



- Ensuring that people writing code have the correct skills and receive appropriate rewards is in *WP5: **Training and Recognition***
- Finally, *WP1*: Framework of the **European Network of Research Software Quality** helps dissemination, connections to the science communities (2-way!) and building the European Virtual Institute for Research Software Excellence as a common endeavour
 - In terms of “advertising ICFA data lifecycle recommendations”, we could have a talk for the EVERSE Network in the future

A brief intro to the RSQKit

- The Research Software Quality Kit (RSQKit) is:
 - A **task driven** guide – **how** to improve research software quality
- It contains:
 - **Tools** related to improving RSQ
 - **Training** related to improving RSQ
 - **Exemplars** of good practice: *Research Software Stories*
- It explains:
 - **Roles** involved in research software
 - Research Infrastructure and **clusters** with heavy use of Research Software
 - How to understand **research software process** for different types of research software

Inspired by



<https://everse.software/RSQKit/>

A product of





















RSQKit audience

- Researchers who code
- Research software engineers
- Those managing and/or procuring funding for project with a large software component
- Those running research infrastructures using software
- Those developing software related policy at organisations and in projects
- pilots (from science clusters) as first testing ground: included in the editorial board



Intended audience **similar / mapped to ICFA lifecycle stakeholders!**

RSQKit tasks, so far

<p> Your tasks</p> <p>Choosing languages, tools & infrastructures</p> <p>How to decide which programming languages, tools and infrastructures to use?</p>	<p> Your tasks</p> <p>Citing software</p> <p>How can people cite your software?</p>	<p> Your tasks</p> <p>Continuous Integration and Continuous Delivery/Deployment</p> <p>How can you use CI/CD in software development?</p>
<p> Your tasks</p> <p>Creating a good README</p> <p>How to create a good README document for software projects?</p>	<p> Your tasks</p> <p>Documenting software</p> <p>How to document your software project?</p>	<p> Your tasks</p> <p>Documenting software using 'Read The Docs'</p> <p>How to use 'Read The Docs' tool for software documentation?</p>
<p> Your tasks</p> <p>Improving environmental sustainability</p> <p>How to measure and improve environmental sustainability of software?</p>	<p> Your tasks</p> <p>Licensing software</p> <p>How to license your software for reuse?</p>	<p> Your tasks</p> <p>Organising software projects</p> <p>How to organise your software project?</p>
<p> Your tasks</p> <p>Packaging & releasing software</p> <p>How to package and release your software for distribution and reuse?</p>	<p> Your tasks</p> <p>Releasing software</p> <p>How to release your software for reuse?</p>	<p> Your tasks</p> <p>Reproducible software environments</p> <p>How to create a development environment for your software so others can run and contribute to your software?</p>
<p> Your tasks</p> <p>Software documentation</p> <p>How to write clear and useful software documentation for developers and end-users</p>	<p> Your tasks</p> <p>Software identifiers</p> <p>How to uniquely identify your software and its versions?</p>	<p> Your tasks</p> <p>Software metadata</p> <p>How to describe your software using metadata?</p>
<p> Your tasks</p> <p>Task automation using GitHub Actions</p> <p>How to set up GitHub Actions on software repositories for task automation</p>	<p> Your tasks</p> <p>Testing software</p> <p>How to test your software?</p>	<p> Your tasks</p> <p>Using version control</p> <p>How to version control your software?</p>

- Common **software tasks** that improve quality, **mapped** to data ICFA lifecycle **recommendations**

- RSQKit also has **research software stories**: the pilots fill a template to showcase their software and journey in EVERSE

- Example for ACTS [here](#)

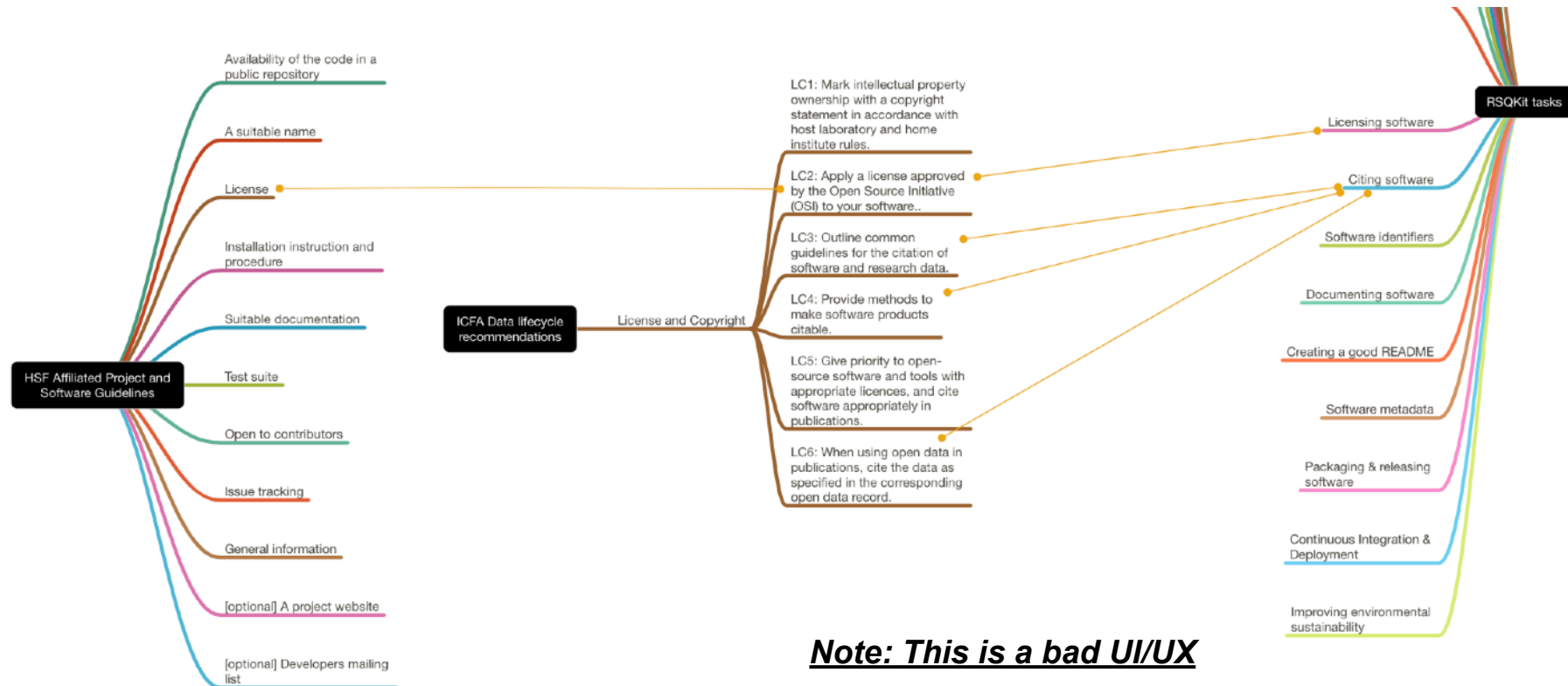


A product of



How does the “mapping” across these efforts look like?

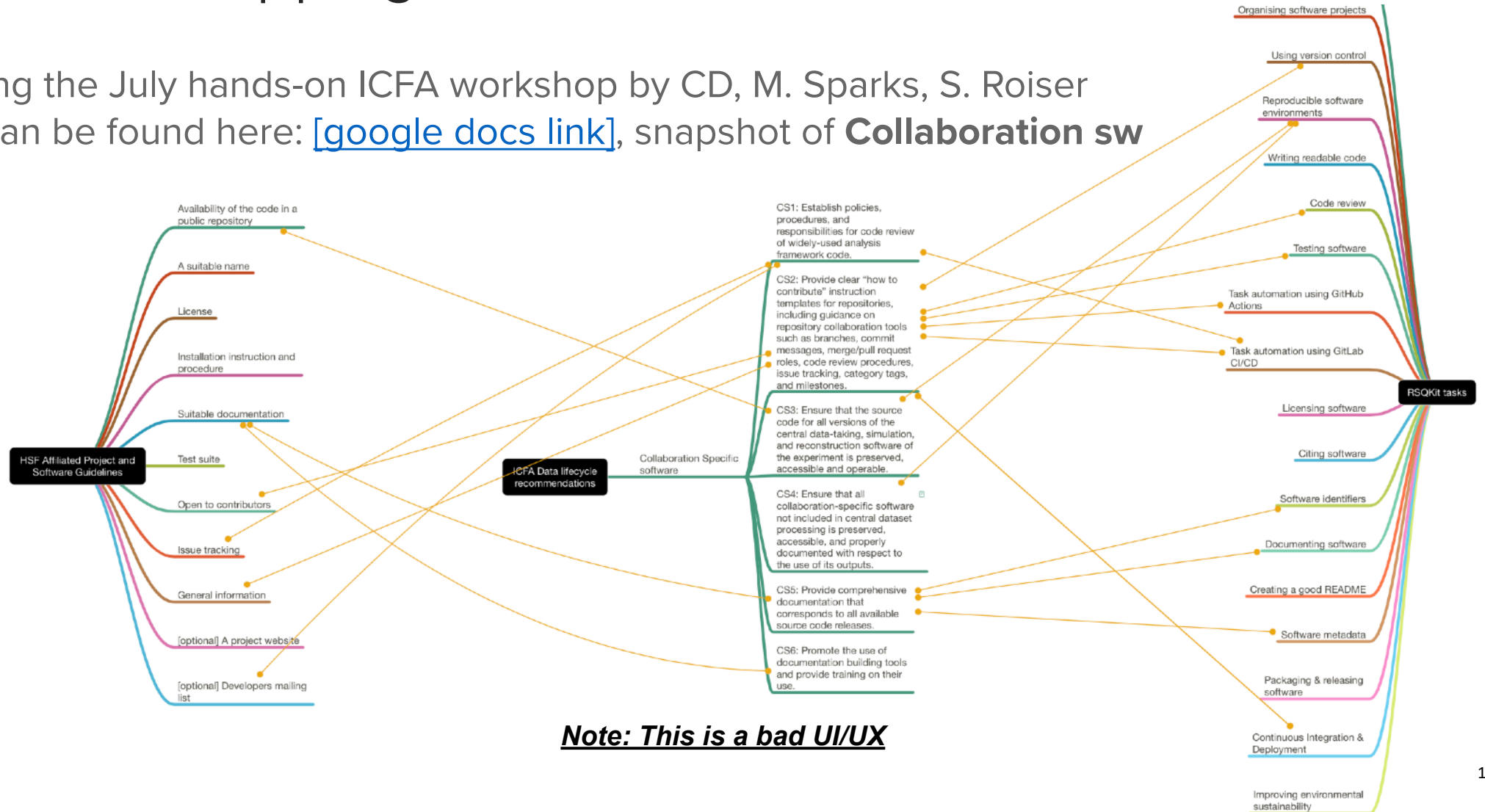
- Work done during the July hands-on ICFA workshop by CD, M. Sparks, S. Roiser
- Full document can be found here: [\[google docs link\]](#), snapshot of **License and Copyright**



Note: This is a bad UI/UX

How does the "mapping" across these efforts look like?

- Work done during the July hands-on ICFA workshop by CD, M. Sparks, S. Roiser
- Full document can be found here: [\[google docs link\]](#), snapshot of **Collaboration sw**



Note: This is a bad UI/UX

Thoughts? And thanks!

Contact: contact@everse.software

EVERSE Network

<https://ec.europa.eu/eusurvey/runner/EVERSENetworkJoinIndividual>

Website:

<https://www.everse.software/>

BlueSky:

<https://bsky.app/profile/eosc-everse.bsky.social>

LinkedIn:

<https://www.linkedin.com/company/eosc-everse/>

FOSSTodon:

https://fosstodon.org/@eosc_everse



**Funded by
the European Union**

This project has received funding from the European Union's Horizon Europe Programme under GA 101129744 – EVERSE – HORIZON-INFRA-2023-EOSC-01-02

