Summary of the Open Science workshop @ ECSS 2026, Porto

(notes taken by Boris Baldassari, summary by Michael Goedicke)

The format of the workshop was to present a few short introductions (see accompanying slides) followed by a round of discussion with the available participants.

Topic 1 – Open source software – The engine of open science in Informatics.

The role of software especially open source software was discussed in academic and scientific processes. Software becomes more important as a scientific tool and needs a proper acknowledgement. For example, it can and should be cited in proper way. There are ways to accomplish this, for example the SWHID[1]. This also entails the availability of an infrastructure to effectively specify and find the related piece of software in an archive and/or repository.

Another move to provide better visibility of software authors in publication processes is to not only list the authors of the publication at hand but add also other roles of contributors as well. This could include the role of a software author and / or data contributors when software and research data is important for the publication.

It is reported that in some (academic hiring) committees require candidates to provide software references. It is a good practice, but it is not enough: not all contributions are equivalent, and not all reviewers can actually make the difference. It would be nice to have a better, more formal way to assess this. This is a difficult topic, as even software practitioners cannot easily judge the quality or relevance of a software asset. The contribution is not necessarily just lines of code; it can be a protocol, interface, an architecture, etc. It is highly desirable to have some tool support to facilitate an transparent and effortless assessment of such artifacts.

In some computer science conferences (e.g. ACM/IEEE International Conference of Software Engineering) progress has been made by applying a badge system for the assessment of the accompanying research software artifact. This still is in phase of a learning curve and more conference could introduce such a scheme.

Topic 2 – Teaching open source – Building skills for the future.

It was first discussed where the differences between non open and open software were. The difference with non-open source software is actually significant. With open source, all the details are available, whereas with proprietary software at best provides a description of the interfaces and this applies only if the documentation is good. Having access to the code, the evolution and dynamics of development, and all artifacts is a game changer for teaching.

What difference does it make to use open source software? Here are some answers:

- The (absence of high) fees make a huge difference. Also, license management is much easier or non existent. This provides a simple basis for the organization of the software developement/ programming courses.

1 c.f. https://www.swhid.org/

© Informatics Europe, 2026, CC BY-SA 4.0

- The availability of additional tools and development ecosystem provides an easy way to provide students with a professional software development experience at low cost. Java as an example, because it's open source, has a rich set of tools, resources for e.g. software quality assessment and beyond.
- In many cases teaching programming does not involve only one specific language or idiom. For example, when functional programming is the topic then not a specific language is in the focus. Thus, one can use whatever language and using open source makes a lot of sense.
- It's hard to think nowadays about a programming language / tool which is not open source and hard pay for such a basic commodity. This is great but, of course, it needs a community to maintain such tools. There are few exceptions of popular language systems which are not open source like Matlab, still.
- Learning from open source engineering i.e. how open source is developed is a great resource for teaching. There are some studies available, see e.g. the work of Johan Linåker[2].

Topic 3 – Infrastructure & Sustainability – Building Ecosystems for OSS

This discussion addressed the infrastructure aspects to maintain a proper environment for teaching and research for the creation and maintaining as well as using open source software.

The popular repositories like GitHub organize software development in addition to being a repository as many have respective IDs to access GitHub. However, being dependent in a single resource will pose a risk and a more distributed and federated approach is needed to circumvent isolation and support a social network for development. While this is desirable it poses currently a few challenges in collaboration. Flexible and effective naming, finding and access schemes are needed.

We would love to see Informatics Europe have an official recommendation about a standard for meta (issues, MRs, discussions, etc.) and forge federation. Also a proposal for federated / unified authentication among a network of forges e.g. based on ORCID would be very desirable. There is work which can be used here e.g. CRUE[3] managing a large set of Spanish universities, or the work of eduroam[4].

2 https://www.linaker.se/
3 https://www.crue.org
4 https://eduroam.org/

© Informatics Europe, 2026, CC BY-SA 4.0