



On the Lack of Recognition of Software Artifacts and IT Infrastructure in Educational Technology Research

Natalie Kiesler ¹ und Daniel Schiffner ²

Abstract: In the context of educational technology research, it is common practice that computer scientists and IT specialists provide support in terms of software and infrastructure for data gathering and processing, storage, analysis and many other services. Ever since Big Data, Learning Analytics and machine learning algorithms have become increasingly feasible, the implementation of programs can be considered a cornerstone of today’s professional research. Contrary to this trend, software as a method for research is hardly recognized within the community, conferences and publication organs. The same applies to processed research data. Therefore, the authors question the current practices and lack of FAIRness related to the publication of software artifacts by discussing the challenges in terms of acknowledgements, review processes, reproducibility and reuse. The paper concludes with recommendations for future FAIR and Open Science practices.

Keywords: Open Science, FAIR principles, epistemology, technology-based research, software, research data management

1 Introduction

Computer scientists and IT specialists experience several challenges when trying to publish research data, software solutions and implementations for data processing. In the context of conference calls for papers, for example, researchers are invited to submit source code, data, survey texts, protocols and other supplementary material that is supposed to “help others replicate your work”. However, these invitations are too often accompanied by the following assertion: “Reviewers are not required to review your supplementary materials, your paper submission must stand on its own.” [Ch22] The inherent dichotomy creates a vicious cycle: Technology-related research contributions can only, if at all, describe software models and architectures on a very abstract level and developers have few incentives to publish high quality source code. As a consequence, challenges to reuse, replicate or re-implement research are imminent in the field [BHR18]. Threats to validity, bias related to the publication of only positive and significant results (File-Drawer problem), as well as high costs of sharing information amplify the problems of non-transparent, closed approaches to research [VR18].

¹ DIPF | Leibniz Institute for Research and Information in Education, Information Center Education, Rostocker Straße 6, 60323 Frankfurt am Main, kiesler@dipf.de, <https://orcid.org/0000-0002-6843-2729>

² DIPF | Leibniz Institute for Research and Information in Education, Information Center Education, Rostocker Straße 6, 60323 Frankfurt am Main, schiffner@dipf.de, <https://orcid.org/0000-0002-0794-0359>

With sustainability as this year's conference theme in mind, the current recognition and publication practice of software artifacts in educational technology research raises several questions. In this paper, the authors discuss the merits of technology for research along with the challenges for implementing, reporting and keeping track of software solutions as part of the epistemological process. As a result, we advocate for recognizing software artifacts as research in alignment with the FAIR principles, thereby easing software reuse and replication.

2 Background and Current Practices

The development of software artifacts and infrastructure is a continuous process that can cause several challenges in research within or utilizing computer science. Although keeping track of software versions with the help of repositories such as GitHub is simple, receiving recognition for software, its versions and components as a tool or methodology for research within the common publication metrics is not. The FAIR approach to research data management requires **F**indable, **A**ccessible, **I**nteroperable and **R**eusable research including data [Go22]. These principles align with the calls for Open Science [Ce22, Wi14], and to some extent with the open access strategy of the German Research Foundation (DFG) [Df22]. In computer science and related disciplines such as educational technology, however, data also translates to software artifacts in the form of scripts, programs or tools that gather or process data to enable experimentation.

In this context, few options for the publication of software artifacts can be considered first steps towards Open Science and FAIRness. The most common example is the use of repositories for code or software modules. Online platforms based on GitHub inherently support version control and collaboration, thereby reflecting on the development of software as a step-by-step process. However, GitHub does not provide artifacts with a persistent Digital Object Identifier and, hence, are not permanently citable in a meaningful way. Zenodo, for example, does, but the threshold for a successful submission (i.e., for it to be reusable) is extremely high in terms of the expected software maturity and documentation. Another challenge when using Zenodo is the lack of alignment with the software development life cycle. Furthermore, there are very few incentives for authors to participate in this process. A second example is the initiation of new categories at conferences aiming at the introduction of tools or other resources. The problem is that these categories usually provide little room for a thorough presentation, e.g., within two (demo) to six (practice) pages at DELFI. And again, these categories neither demand supplementary material, nor their review. This is different in other CS domains such as software engineering, where conferences comprise technical tracks up to ten plus two pages, open science policies and compulsory quality reviews. A third approach towards software publication is the *NISO RP-31-2021 Reproducibility Badging and Definitions* as recommended practice [Ni22]. It aims at a standardized badging system that can be applied in review processes of software artifacts. Moreover, the recommended practice outlines badge systems for sharing and reviewing data and methods which are currently applied by publishers and professional societies including ACM and IEEE. A similar approach is

given by the CESSDA Technical Guidelines [Co21]. The last good practice example is constituted by the transfer of FAIR principles to research software [KGH21]. This effort highlights the role of software in the research process and the need for more transparency and data provenance.

Despite the few good practice examples and an increasing awareness of software artifacts' role in Open Science, we observe the following deficits of current practices in terms of FAIRness and the four guiding principles [Gi22]:

- **F**: Lack of strategies for long-term preservation and adding metadata, as well as a lack of search engines, options and infrastructure for publication.
- **A**: Proprietary software components and protocols are unavailable and not universally-implementable.
- **I**: Lack of archives for environments/contexts for successful execution of software artifacts (e.g., virtual machines, containers, formats or versions). Complex systems are not easily set up or embedded in a research environment.
- **R**: Lack of standards for data provenance and documentation in machine-readable formats.

3 Discussion of the Current Practice

The outlined deficits in terms of FAIRness imply that present approaches are neither sufficient for the thorough development, investigation, nor the reuse and replication of software artifacts in the context of educational technology research. Time and effort required for the implementation of FAIR are not recognized, and due to the lack of incentives it is impossible to improve the current recognition practice of software artifacts. This is why we need to discuss the merits of technology for research, its epistemological contribution, and how we can adequately report and publish software artifacts to ease verification, reproducibility and reuse of software artifacts.

It is interesting that the awareness of quantitative and qualitative research methods and the corresponding paradigms seems widespread among researchers of all disciplines, whereas the development of algorithms and software as research methods or outcomes is not. Similarly, software is data, but there is more to it. Considering this a major desiderat in the community, we need to become more explicit about what we perceive as research. Otherwise, the lack of recognition of and publication formats for software artifacts will continue to lead to several challenges for both scientists and developers. In the educational technology and computing education context, it seems as if there is still little conscientiousness for the contribution of software artifacts in the epistemological process [KMM10]. The development of tools as a research area is described as challenging “both for designing and reporting research” [Ma14]. In an effort to identify research purpose dimensions in computing education research, Malmi et al. [Ma10, p.5] distinguish 12 categories and three clusters: descriptive, evaluative and formulative. Five of these categories directly relate to information systems, technologies, or the modeling and problem solving via algorithms. Even though their study implies education research's

character and its focus on tools and technologies, review processes, open repositories, persistent identifiers and other instruments for the implementation of FAIRness, software reuse and replication studies are hardly evident.

Reuse and re-implementation of software often becomes impossible without source code, IT infrastructure and data, and yet research papers keep on describing their methodology in natural language. Even if links to repositories and other resources that would help understand the research process are provided, they are not a crucial component of the review process. However, checking code for what it does, reviewing its implementation, quality and results should be the standard procedure among publication organs. After all, this is the only method that allows for verification, supports comprehensibility, and finally helps foster software reuse, replication and its further investigation among developers and researchers. It is tempting to assume that the lack of research on tools or software artifacts in educational contexts is partially due to the lack of reproducibility of this type of data.

The low numbers of publications on or including tools may also be due to the rapid advances in computing and the community's concerns that arise from it. In general, the threshold for the publication of software is extremely high and takes time: It requires a certain level of maturity, a proper documentation and adhering to a good style, e.g., naming conventions. Implementing the CESSDA Technical Guidelines [Co21], for example, demands extensive resources, and thus time and effort to achieve high quality software. By the time the software artifact and/or research has achieved the level to get published, the developments may already be deprecated. The fact that this effort is not accounted as a scientific contribution within the traditional research/publication metrics makes it even worse. As a consequence, the reluctance to publish software is reinforced.

Yet another challenge related to software as research data is the necessity of keeping track of its versions, further developments and the respective results, thereby recognizing the contributions of individuals and the software's history. Despite the availability of open repositories such as GitHub, uploading software artifacts alone is not a solution ensuring data provenance. It neither provides credit to the authors in the form of persistent identifiers. Moreover, software artifacts do not work in an isolated repository without a proper description of the context required for its successful execution in the long term. Software artifacts and their versions must be accompanied by data related to the study protocol, instruments, data gathering and analysis, the executing infrastructure, as well as results. Therefore, tracing all steps of the research process and software versions is still a challenge. It is thus important to allow for an integrated presentation of all of these elements and their dependencies to allow for reuse (and reproducibility). This is supposed to be the standard for any research, regardless of whether or not software was developed or utilized.

It is true though that educational technology research differs from empirical research in education where replication studies are considered a valid construct and goal of research. In the context of technology-based research, software replications can and should be avoided. Regardless of this divergence, we should be able to understand and verify software artifacts based on the published material and source code. In an ideal scenario,

tools and technologies can be reused, improved or extended through collaborative efforts so that the wheel will not be reinvented again and again.

4 Conclusion and Next Steps

Having discussed the systematic challenges resulting from the lack of recognition towards software artifacts, we suggest several steps for the educational technology research community to become more FAIR and adhere to the Open Science principles:

1. Provide attribution in the scientific community for developed infrastructures and software solutions. They are not just a tool; they are the foundation of empirical research, method and result within the context of the technology-enhanced learning community.
2. Foster existing methods to publish software and encourage extension of or alternatives to existing methodologies (e.g., OSF, Zenodo, LearnSphere, GitHub, Figshare, Renku, Software Heritage Foundation, etc.).
3. Establish new publication categories that reflect the software development processes, including mandatory reviews of software artifacts and infrastructure among publishing organs.
4. Live and preach: “Don't reinvent the wheel”. The FAIR principles, among others, help overcome restrictions, but are only guiding towards an Open Science community. If FAIR was the reality, other community members could use existing solutions and contribute to them without having to invest into a large ecosystem. While software constitutes the starting point, interfaces and services can be considered the gold standard we should achieve.

In summary, there is a need for incentives and attribution within the community when it comes to software and IT infrastructure publication. There should be persistent identifiers and accompanying publication formats. More importantly, software, IT infrastructure and source code must be recognized as output of scholarly research [JHK21], and common guidelines for its review must be established. We strongly encourage further initiatives to improve software quality and interoperability within our community.

Bibliography

- [BHR18] Beardsley, M.; Hernández-Leo, D.; Ramírez, R.: Seeking reproducibility in multimodal learning experiments: Assessing an EEG study of the testing effect. *Journal of Computer Assisted Learning*, 2018, <https://doi.org/10.1111/jcal.12265>.
- [Ce22] Center for Open Science, <https://www.cos.io/initiatives/top-guidelines>, Accessed: 18.2.2022.
- [Ch22] CHI Conference on Human Factors in Computing Systems, <https://chi2022.acm.org/for-authors/presenting/papers/>, Accessed: 17.02.2022.

- [Co21] Consortium of European Social Science Data Archives, <https://docs.tech.cessda.eu/>, Accessed: 04.03.2022.
- [Df22] Deutsche Forschungsgemeinschaft, https://www.dfg.de/foerderung/programme/infrastruktur/lis/open_access/, Accessed: 18.2.2022.
- [Gi22] Gesellschaft für Informatik e.V., <https://youtu.be/jLcYCgHxI8s>, Accessed: 22.2.2022.
- [Go22] GO Fair, <https://www.go-fair.org/fair-principles/>, Accessed: 10.2.2022.
- [JHK21] Jay, C.; Haines, R.; Katz, D.S.: Software must be recognised as an important output of scholarly research, *International Journal of Digital Curation*, v.16(1), 2021.
- [KGH21] Katz, D. S.; Gruenpeter, M.; Honeyman, T.: Taking a fresh look at FAIR for research software, *Patterns* v.2(3), 100222, 2021.
- [KMM10] Kinnunen, P.; Meisalo, V.; Malmi, L.: Have we missed something? identifying missing types of research in computing education. In: *Proceedings of the Sixth international workshop on Computing education research*, ACM, New York, p. 13-22, 2010.
- [Ma14] Malmi, L.: Tools research-what is it? *ACM Inroads* 5, 3 (09/2014), p. 34–35, 2014.
- [Ma10] Malmi, L.; Sheard, J. S.; Bednarik, R.; Helminen, J.; Korhonen, A.; Myller, N.; Sorva, J.; Taherkhani, A., Characterizing research in computing education: a preliminary analysis of the literature. In: *Proceedings of the Sixth international workshop on Computing education research*. ACM, New York, p. 3-12, 2010.
- [Ni22] National Information Standards Organization, <http://www.niso.org/standards-committees/reproducibility-badging>, Accessed: 18.2.2022.
- [VR18] van der Zee, T.; Reich, J.: Open Educational Science. *SocArXiv Papers*, <http://doi.org/10.17605/OSF.IO/D9BME>, Accessed: 18.02.2022.
- [Wi14] Winfield, A. F.: Open science-a three level approach. *Science, Innovation and Society-Responsible Research and Innovation Conference*, Rome, 20.11.2014.