SBC Computing Reviews, 2026, 06:1, Special Issue on Software Engineering
ISSN: 2966-3938 • doi: 10.5753/reviews.2026.7212
This work is licensed under a Creative Commons Attribution 4.0 International License.

Review

Beyond the FOSTER Taxonomy: an Extended Review of Open Science-Related Practices, Evidence, and Reproducibility in Software Engineering

André F. R. Cordeiro [State University of Maringá (UEM) | *cordeiroandrefelipe@gmail.com*]
Edson OliveiraJr [State University of Maringá (UEM) | *edson@din.uem.br*]

Informatics Department, State University of Maringá, Av. Colombo, 5790 - Bloco C56, Maringá, PR, 87020-900, Brazil.

Abstract. Context: Open Science is increasingly recognized as essential for improving transparency, reproducibility, accessibility, and collaboration across scientific domains. However, its adoption within Software Engineering has evolved in a fragmented manner, with practices often limited to isolated initiatives such as open data portals, ad hoc artifact sharing, or descriptive methodological reporting. The field lacks consolidated evidence about how Open Science principles are interpreted, operationalized, and sustained in Software Engineering research. **Objective:** This study aims to characterize how Open Science principles, practices, and infrastructures have been addressed in Software Engineering research and to identify gaps, challenges, and opportunities that shape its current and future development. The goal is to provide both a descriptive account of existing efforts and a prescriptive foundation for strengthening openness in the field. **Methodology:** We conducted a Scoping Review of 15 primary studies, systematically analyzing Open Science practices, challenges, topics, contexts, empirical robustness, alignment with UNESCO principles, and the extent to which studies support reproducibility through availability, completeness, traceability, and verifiability of research artifacts. **Results:** The review reveals uneven and often partial adoption of Open Science principles. Transparency and accessibility appear more frequently, particularly in studies involving open data infrastructures or methodological reporting, whereas reproducibility, collaboration, empirical rigor, and the availability of comprehensive artifacts remain limited. Evidence is heterogeneous, with only a subset of studies providing robust or well-documented empirical foundations. **Main Contributions:** The study delivers a structured set of observations, an integrative research agenda, and a set of cross-cutting and prospective actions spanning researchers, educators, conferences, journals, infrastructure providers, and policymakers. These contributions offer actionable guidance for improving artifact quality, strengthening socio-technical integration, enhancing evaluation practices, and supporting sustainable adoption of Open Science in Software Engineering. **Conclusions:** Although encouraging progress is visible, Open Science in Software Engineering remains fragmented and inconsistently implemented. Advancing the field requires coordinated community efforts, stronger empirical foundations, improved infrastructures for sharing and preservation, broader educational support, and sustained commitment to reproducibility, transparency, and collaborative knowledge production.

Keywords: Open Science, Software Engineering, Openness, Open Research, Practices, Artifacts, Observations, Reproducibility, Transparency, Accessibility, Open Data, FAIR Principles, UNESCO Principles, Socio-technical Factors, Open Platforms, Scoping Review.

Received: 13 January 2026 • **Accepted:** 19 August 2026 • **Published:** 30 September 2026

1 Introduction

The ISO/IEC/IEEE vocabulary defines Software Engineering (SE) as the application of scientific and technological knowledge to software development [Standard, 2017]. SE has evolved over the last few decades, driven by scientific and technological research. Its high impact and social contributions support this evolution.

The SE area has incorporated various empirical research methods from fields such as Medicine and Social Sciences [Lorey *et al*., 2022; Ralph *et al*., 2016]. These rigorous methods facilitate knowledge sharing in SE research, including systematic reviews [Kitchenham and Charters, 2007] and experiments/quasi-experiments [Wohlin *et al*., 2012].

SE produces, uses, and reuses different artifacts. Examples of artifacts include source code, binary code, scripts, software, configuration files, images, databases, documentation, datasets, and licenses [Pfeiffer, 2020; Timperley *et al*., 2021; Frachtenberg, 2022]. Additionally, artifacts may include communicability directives [Lopes *et al*., 2020], tools related to reproducibility [Anchundia *et al*., 2020], program execution models, data flow models [Dennis, 2017], containers [Cito and Gall, 2016], methodologies [González-Barahona and Robles, 2012], models and requirements [Nicolás and Toval, 2009], and conceptual models [Flint, 2009].

Creating or using an artifact is not a trivial task [Pfeiffer, 2020; Basili *et al*., 2007]. Issues arise during an artifact's elaboration or use, such as difficulty finding specific information, broken links, and a lack of standardization [Méndez Fernández *et al*., 2019a; Ghazi and Glinz, 2017]. Different studies in the literature present observations for managing artifacts in SE [Lopes Damian *et al*., 2021; Lopes *et al*., 2020; Méndez Fernández *et al*., 2019a; Ghazi and Glinz, 2017; Grechanik *et al*., 2014; Krishnamurthi, 2013; Basili *et al*., 2007].

The global Open Science (OS) movement, primarily

spearheaded by UNESCO [UNESCO, 2022], promotes access to research and its outcomes for all individuals. It has examined various issues across different domains, including reproducibility [Mauerer and Scherzinger, 2022], experimentation [Baltes and Ralph, 2022], design science methodologies [Engström *et al*., 2020], and education [Farnham *et al*., 2017]. These efforts aim to enhance and open up research project resources and discoveries.

The OS movement is gaining traction in SE, as evidenced by growing interest within the community. This interest is evident in the adoption of OS practices by the Empirical Software Engineering Journal and at conferences such as the International Conference on Software Engineering (ICSE), International Symposium on Empirical Software Engineering and Measurement (ESEM), Mining Software Repositories (MSR), International Conference on Software Analysis, Evolution and Reengineering (SANER), International Conference on Software Analysis, Evolution and Reengineering (SANER), and the Brazilian OpenScienSE workshop. Over the past few years, several OS policies for SE have been developed and utilized [Mendez *et al*., 2020; Ernst and Baldassarre, 2023].

The OS movement encompasses various related concepts and practices that aim to make research accessible to all individuals [Pontika *et al*., 2015]. This paper defines these concepts and practices and identifies the challenges for OS in SE. Given the emerging nature of this area in SE, we conducted a Scoping Review Study (SRS) [Khalil and Tricco, 2022; Munn *et al*., 2018]. Based on the SRS findings, we proposed several observations for OS in the coming years.

Finally, this paper is an extended and substantially revised version of our previous study published at SBES 2025 [Cordeiro and OliveiraJr, 2025]. The original research reported initial findings from a Scoping Review aimed at identifying Open Science (OS) concepts, practices, and challenges in Software Engineering, providing a primarily descriptive overview constrained by conference length limitations. This extended version significantly broadens and deepens that contribution by incorporating additional research questions, expanding the analytical scope, and offering a more rigorous, multidimensional synthesis. In particular, this paper introduces new analyses concerning the empirical robustness of the primary studies, their alignment with UNESCO Open Science principles, and a systematic assessment of reproducibility support grounded in established artifact-evaluation guidelines. Moreover, the results are substantially enriched through deeper cross-study comparisons, new tabular and visual syntheses, and an integrative discussion that culminates in a consolidated research agenda and actionable recommendations for researchers, educators, venues, infrastructure providers, and policy-makers. We have rewritten and reorganized large portions of the manuscript to strengthen the theoretical grounding, improve coherence, and provide a more comprehensive and prescriptive contribution than the earlier conference version.

This paper is organized as follows: Section 2 introduces essential concepts on SE, artifacts, and OS and discusses related work; Section 3 describes the methodology adopted for our SRS; Section 4 presents the SRS and its results based on the established research questions; Section 5 discusses the obtained results and provides several observations based on current and prospective actions of OS for SE; Section 6 discusses the threats associated with this SRS; Section 7 presents a set of prospective actions that the Software Engineering community can adopt based on this study; and Section 8 concludes this paper.

2 Background and Related Work

In this section, we present concepts that are essential to understanding our work and the related discussions.

2.1 Open Science

According to UNESCO [2022], OS is a set of principles and practices that aim to make scientific research from all fields accessible to everyone for the benefit of scientists and society as a whole.

Various OS-related initiatives are observed across studies. One initiative described in principles, processes, and badges is presented by Méndez Fernández *et al*. [2019b]. Understanding OS at an advanced level requires understanding the concept of openness. This concept can be discussed in terms of transparency, access, participation, and democracy [Schlagwein *et al*., 2017].

When analyzing the OS movement, several initiatives can be mentioned, such as those in Mosconi *et al*. [2019] and Damasceno *et al*. [2021], which focus on artifact sharing.

The literature presents various OS taxonomies [Silveira *et al*., 2023]. Pontika *et al*. [2015] present a taxonomy of possible subareas of OS research. The European Commission developed the Facilitate OS Training for European Research (FOSTER) taxonomy [Open Science, 2022].

Among FOSTER's objectives is qualifying various stakeholders, including researchers, grantees, and repository managers. Figure 1 presents the FOSTER taxonomy.

Such a taxonomy includes essential branches: Open Access, Open Data, Open Reproducible Research, OS Evaluation, OS Policies, and OS Tools.

Investigations in the context of OS are gaining attention in the SE community OliveiraJr *et al*. [2024]; Mendez *et al*. [2020]. These studies address different aspects, including justifications for OS adoption, observations, challenges, and potential issues.

2.2 Open Science in Software Engineering

The development, use, and reuse have long been considered in the field of SE [Sommerville, 2016]. This is evident in studies and practices involving Software Reuse [Frakes and Kang, 2005; Krueger, 1992] and Experimental Packages [Wohlin *et al*., 2012; Juristo and Gómez, 2010]. Even with this background, the need to discuss how this knowledge applies to all artifacts generated or used in SE projects became apparent. In this context, OS can be considered. The adoption of OS-related principles and practices in SE is only beginning to be addressed by Mendez *et al*. [2020] and OliveiraJr *et al*. [2024].

The SE community has undertaken various initiatives and studies within the context of OS. In Méndez Fernández *et al*. [2019a], the Empirical Software Engineering Journal (EMSE) initiative involves principles, actions, and badges. The International Conference on Software Engi-

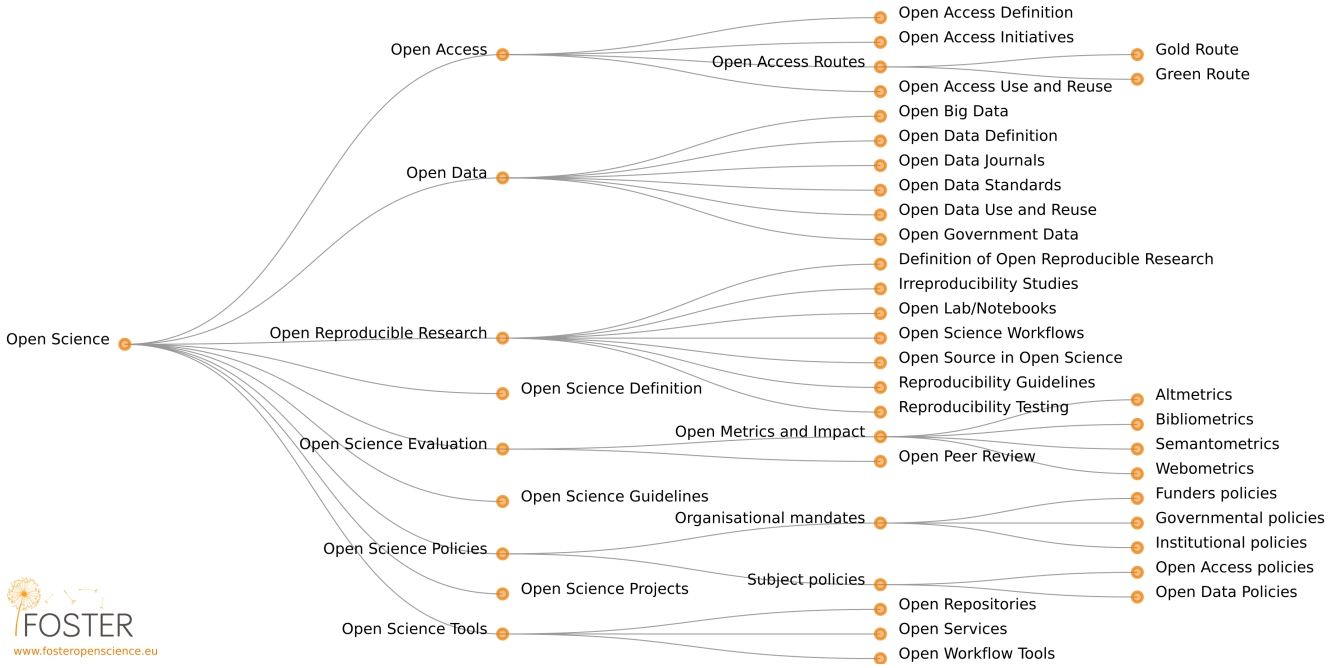


Figure 1. Open Science Taxonomy by FOSTER [Foster, 2022].

neering (ICSE) and other leading international SE conferences have issued calls for artifact submissions for evaluation. The International Conference on Software Analysis, Evolution, and Reengineering (SANER) features a specific track, called REproducibility Studies and NEgative Results (RENE), aimed at promoting the reproducibility of previous study results and publishing studies with negative or null results.

In Brazil, the Workshop on Open Science Practices for Software Engineering has been held in recent years to exchange experiences on OS practices and challenges. OS initiatives also consider organizations and infrastructures, such as OpenAIRE [Manghi *et al.*, 2019], Figshare [Figshare, 2024], and Zenodo [European Organization For Nuclear Research and OpenAIRE, 2013].

2.3 Related Work

Vicente-Saez and Martinez-Fuentes [2018] performed a systematic literature review to establish an integrated definition of OS. They analyzed 75 papers and classified OS descriptions into knowledge, transparent knowledge, accessible knowledge, shared knowledge, and collaboratively developed knowledge. Such a work complies with the Open Science Framework (OSF)[1] taxonomy in many of the terms presented.

One Systematic Mapping of Open Data presents a classification and analysis of the studies conducted by the scientific community between 2006 and 2019 [Enríquez-Reyes *et al.*, 2021]. The analysis considered publication, impact, subject matter, domain, life cycle, and research type.

These studies review and map OS-related topics, but none address the SE area.

[1] https://osf.io/

3 Research Methodology

In this work, we conducted a Scoping Review Study (SRS) [Khalil and Tricco, 2022; Munn *et al.*, 2018] of the literature.

Our goal is to identify primary studies that report or directly investigate OS-related practices, challenges, topics, and contexts in SE. We based our SRS on explicit and implicit practices identified in the primary literature sources presented in Section 2.1 and in the FOSTER Taxonomy.

We opted for a Scoping Review to broadly characterize OS and related practices as an emerging research topic in SE. The association between OS and SE is evident in studies that address one or more subareas of OS, as described in the FOSTER taxonomy.

4 The Scoping Review Study

This section presents the results of the performed Scoping Review Study (SRS).

4.1 Aim and Research Questions

This SRS is **aimed at analyzing** OS, **with the purpose** of characterizing the current implicit and explicit OS-related practices in the SE area, **from the point of view of** SE researchers, **in the context of** primary studies considering OS and SE. Therefore, we defined the following research questions:

- **RQ1: What explicit or implicit practices associated with OS have been observed in SE?** It aims to identify OS practices used in SE and understand the benefits associated with them.
- **RQ2: What are the observed challenges in applying OS practices in SE?** We aim to examine the current challenges associated with implementing the identified OS practices in SE. Our interest in these challenges stems from the understanding that they could impede, or even render unworkable, the adoption of OS practices

in SE projects.

- **RQ3: What OS topics have been most considered in SE?** We aim to identify the OS subareas most explored in SE. This information may inform future studies on OS-based SE practices.
- **RQ4: In what contexts it is observed practices associated with OS in SE?** It aims to understand the contexts that have facilitated the implementation of OS-related practices in SE. Based on research conducted in OS, we recognize that these practices are relevant to both software development and SE research.
- **RQ5: What empirical evidence supports the OS-related practices and propositions reported in the selected studies?** This question assesses the empirical robustness of the studies, characterizing the depth of validation and methodological rigor.
- **RQ6: How do the OS-related contributions reported in SE align with the UNESCO Open Science principles (transparency, accessibility, reproducibility, collaboration)?**
  This brings an internationally recognized lens beyond FOSTER's internal classification.
- **RQ7: To what extent do the selected studies enable reproducibility considering availability, completeness, traceability, and verifiability criteria?** Based on the ACM Artifact Guidelines and ESEM/ICSE recommendations.
- **RQ8: What research gaps, inconsistencies, and opportunities emerge when the selected studies are analyzed collectively?** Producing a structured research agenda for OS in SE.

For this SRS, we asked two experienced SE researchers and research artifacts evaluation experts to assess the protocol, research questions, and procedures.

The two SE researchers have professional and academic experience, particularly in planning and conducting secondary studies. Two other OS experts have experience managing, sharing, and evaluating artifacts. They act as artifact reviewer in the ICSE Artifact Evaluation track.

4.2 Search Process

We selected and tuned the search terms based on the FOSTER taxonomy and the work of Mendez *et al*. [2020]. Consequently, the search strategy was composed of two main concepts:

- **software engineering**; and
- **open science** – "open science", "open access", "open data", "open reproducible research", "open science evaluation", "open science policies", and "open science tools".

The resulting conceptual search string is presented in Table 1.

Although a single conceptual search string was defined, executing it required database-specific adaptations because of differences in query syntax, supported search fields, Boolean operators, and filtering mechanisms. These adaptations were purely operational and necessary to ensure the same search strategy could be executed across all selected digital libraries.

Table 1. General conceptual search string.

"software engineering"
AND
("open science" **OR** "open access" **OR** "open data" **OR** "open reproducible research" **OR** "open science evaluation" **OR** "open science policies" **OR** "open science tools")

To maximize consistency across heterogeneous search interfaces, the search strategy was adapted to each database's capabilities. Minor syntactic adjustments (e.g., insertion or removal of parentheses, quotation marks, or field operators) were performed only when required by the corresponding search engine. Table 2 summarizes the search fields adopted and the operational adaptations performed for each digital library.

Because search capabilities differed across digital libraries, we subsequently performed backward and forward snowballing in accordance with the guidelines proposed by Wohlin [2014] and Badampudi *et al*. [2015]. This complementary strategy was specifically adopted to increase search coverage and identify potentially relevant primary studies that might not have been retrieved during the initial database search due to differences in indexing policies or supported search fields. The complete search protocol, including the exact search strings executed in each digital library, search fields, applied filters, execution dates, and retrieved results, is publicly available in the online replication package.

The online replication package includes the complete search protocol, the exact search strings used in each digital library, the applied filters, the fields searched, the execution dates, the retrieved results, and the intermediate datasets generated during the selection process.

To mitigate the risk of missing relevant studies due to differences in search capabilities across digital libraries, the automated search was complemented by backward and forward snowballing, following the guidelines proposed by Wohlin

Table 3 summarizes the search results per source. The initial set comprises 504 studies.

4.3 Selection Process

The selection process began by defining inclusion (I) and exclusion (E) criteria for the primary studies. Subsequently, we established the following criteria:

- **Inclusion criteria:**
  - **I.1** mention of one or more practices or concepts associated with OS in SE
  - **I.2** explicit mention of one or more challenges associated with OS in SE
- **Exclusion criteria:**
  - **E.1** study that is not in the field of SE
  - **E.2** non-explicit mention of one or more practices associated with OS in SE
  - **E.3** non-explicit mention of one or more challenges associated with OS in SE
  - **E.4** study not written in English
  - **E.5** study with four pages or less

Table 2. Database-specific implementation of the search strategy.

Digital Library	Search Field(s)	Syntax Adaptation	Operational Differences
IEEE Xplore	Document Title	Parentheses removed	Search executed using the document title field according to the platform syntax.
ACM Digital Library	Title	Parentheses removed	Search executed using the title field according to the platform syntax.
SpringerLink	Title contains	Quotation marks and parentheses adjusted	Search adapted to the syntax required by the platform.
ScienceDirect	Title, Abstract, Author Keywords	None	The original conceptual search string was directly supported by the search engine.
Wiley Online Library	Title	Parentheses inserted	Search adapted to the syntax required by the platform.
Engineering Village	Title	Original parentheses preserved	Search executed according to the syntax supported by the platform.
Scopus	Article Title	None	Search executed using the article title field defined in the review protocol.
Web of Science	Title	Original parentheses preserved	Search executed according to the syntax supported by the platform.

Table 3. Summary of the Search Process

Sources	Returned Studies
ACM (Digital Library)	398
Springer	90
ScienceDirect	5
Scopus	4
Engineering Village	3
Web of Science	3
IEEE	1
Wiley	0
Total	**504**

- **E.6** study not published in conferences or journals
- **E.7** duplicate study
- **E.8** study unavailable even after contacting the authors
- **E.9** secondary or tertiary study

We screened all 504 primary studies using the inclusion and exclusion criteria applied to their titles, keywords, and abstracts. A summary of the resulting papers from this process is shown in Table 4.

Table 4. Summary of the Selection Process.

Sources	Returned	Selected	Duplicated	Rejected
ACM	398	8	2	390
IEEE	1	0	0	1
Scopus	4	0	1	3
Springer	90	3	0	87
ScienceDirect	5	2	0	3
Wiley	0	0	0	0
Engineering Village	3	1	0	2
Web of Science	3	0	0	3
Total	**504**	**14**	**3**	**487**

from the search process (Section 4.2).

We also performed forward and reverse snowballing [Badampudi *et al.*, 2015; Wohlin, 2014]. We identified a second set of studies for analysis. We applied the same selection criteria to this second set, initially retrieving 80 studies without duplication. After applying the selection criteria, only one study met them. Finally, we selected 15 primary studies, listed in Table 5.

4.4 Extraction Process

We initiated the extraction process using the 15 primary studies previously selected.

Table 5. Final Set of Studies.

ID	Ref.	Title	Pub. Venue	Year
S01	[Correa *et al.*, 2018]	Investigating Open Data Portals Automatically: A Methodology and Some Illustrations	dg.o	2018
S02	[Aaltonen *et al.*, 2014]	From Mashup Applications to Open Data Ecosystems	OpenSym	2014
S03	[Cordasco *et al.*, 2018]	A Layered Architecture for Open Data: Design, Implementation and Experiences	ICEGOV	2018
S04	[Traverso-Ribón *et al.*, 2013]	Open Data Framework for Sustainable Assessment in Software Forges	WIMS	2013
S05	[Söderberg *et al.*, 2021]	Open Data-Driven Usability Improvements of Static Code Analysis and Its Challenges	EASE	2021
S06	[Kuhn, 2012]	IDEs Need Become Open Data Platforms (as Need Languages and VMs)	TOPI	2012
S07	[Brito *et al.*, 2014]	Brazilian Government Open Data: Implementation, Challenges, and Potential Opportunities	dg.o	2014
S08	[El Bacha *et al.*, 2017]	Exploiting Open Data to Improve the Business Intelligence	BDCA	2017
S09	[Ernst *et al.*, 2021]	Understanding peer review of software engineering papers	EMSE	2021
S10	[Graziotin *et al.*, 2014]	A framework for systematic analysis of open access journals and its application in software engineering and information systems	Scientometrics	2014
S11	[Timperley *et al.*, 2021]	Understanding and improving artifact sharing in software engineering research	EMSE	2021
S12	[Ruiz-Rube *et al.*, 2015]	A framework for software process deployment and evaluation	IST	2015
S13	[Azpeitia *et al.*, 2020]	Volunteering for Linked Data Wrapper maintenance: A platform perspective	Inf. Sys.	2019
S14	[Jagtiani and Lewis, 2016]	Enhancing software engineering curricula by incorporating open, data-driven planning methods	ASEE	2016
S15	[Olsson and Bosch, 2014]	From Opinions to Data-Driven Software R&D: A Multi-case Study on How to Close the 'Open Loop' Problem	EUROMICRO	2014

We read all the papers and extracted the following metadata: title, source, publication venue, and publication year. Additionally, we identified practices, challenges, OS-related topics, and context.

4.5 Results

This section presents the results obtained with our SRS (Table 5).

4.5.1 Metadata Results

The publications span the years 2012 through 2021. In 2014, 2021, and 2018, we have four, three, and two studies, respectively. Each remaining year has one study. This fact indicates a lack of studies in this period, which is expected for such a new topic in SE.

Table 6 summarizes the contribution of each study to each research question.

All 15 studies contribute to RQ1, RQ3, RQ4, RQ5,

Table 6. Contributor Studies per Research Questions.

ID	RQ1	RQ2	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	Count
S01	X	X	X	X	X	X	X	X	08
S02	X	X	X	X	X	X	X	X	08
S03	X	X	X	X	X	X	X	X	08
S04	X	—	X	X	X	X	X	X	07
S05	X	X	X	X	X	X	X	X	08
S06	X	X	X	X	X	X	X	X	08
S07	X	X	X	X	X	X	X	X	08
S08	X	—	X	X	X	X	X	X	07
S09	X	—	X	X	X	X	X	X	07
S10	X	X	X	X	X	X	X	X	08
S11	X	X	X	X	X	X	X	X	08
S12	X	X	X	X	X	X	X	X	08
S13	X	X	X	X	X	X	X	X	08
S14	X	—	X	X	X	X	X	X	07
S15	X	—	X	X	X	X	X	X	07
Count	**15**	**10**	**15**	**15**	**15**	**15**	**15**	**15**	—

RQ6, RQ7, and RQ8, which address OS practices, topics, contexts, empirical evidence, alignment with UNESCO OS principles, reproducibility, and research gaps, respectively. As for RQ2, five studies do not explicitly address the challenges of applying OS practices in SE.

4.5.2 RQ1 - Practices Related to Open Science

Table 7 presents the observed practices in the final set of primary studies.

Table 7. Open Science-related Practices in Software Engineering.

OS Practices	Solution	Study ID	Practice ID
Open Data	software portal	S01	Pr.01
	open data mashups for data-as-a-service approach	S02	Pr.02
	architectural model based on open data	S03	Pr.03
	open data model	S04	Pr.04
	reuse of open data; usage of open collaborative data	S05	Pr.05
	open data portals for software development with different data sets	S07	Pr.06
	Linked Open Data and Ontologies	S08	Pr.07
	Linked Open Data for integrating data hosted by different tools	S12	Pr.08
	Linked Open Data for use or development of Linked Data Wrappers (LDWs)	S13	Pr.09
	software development based on data	S15	Pr.10
Open Platforms	uniform data structures; open architectures	S06	Pr.11
	open data clouds (DBPedia, Wikitology, Yago, WordNet, and others)	S08	Pr.12
Open Review	exemplar process guideline	S09	Pr.13
Open Access	framework for the analysis of open access journals	S10	Pr.14
Artifact Sharing	a plan for creating and sharing artifacts	S11	Pr.15
Open Source	leverage of Open Source Software communities	S14	Pr.16

Several practices are observed in the context of **Open Data**. Regarding the software portal (S01, S07), open data is stored once it is established and can be accessed by different applications. The practice of open data portals for software development, using various datasets, suggests that development portals can adopt an architecture comprising three main elements: extractor, analyzer, and presenter. The extractor retrieves all the necessary data for the software from various sources. The analyzer evaluates the retrieved information and selects only the most important data for users. The presenter generates informative pages and presents them to users.

S02 introduces the concept of open data mashups for the data-as-a-service approach. Mashups can be developed as services. Various access technologies, such as Web Feeds, Web Interfaces, Representational State Transfer (REST), Simple Object Access Protocol (SOAP), and Application Programming Interfaces (APIs), can be considered.

An architectural model based on open data (S03) considers the layers of open data, open information, and open knowledge. Each layer is associated with specific operations, which can be accessed via service APIs. The data layer handles storage, catalog generation, and queries. The open information layer facilitates discussions, visualization, and co-creation of knowledge. The open knowledge layer focuses on sharing operations.

The open data model emphasizes considering key entities within their respective contexts. In the scenario outlined in S04, the model considered various entities, such as projects, datasets, users, and metrics. The reuse of open data and the utilization of open, collaborative data can be viewed from multiple angles, as discussed in Section 5. Data-driven software development can be structured using a hypothesis-based model. The model presented in S15 encompasses activities such as resource backlog generation, resource selection and specification, implementation and instrumentation, gap analysis, hypothesis generation and selection, and alternative implementation.

The level of detail for using or developing Linked Data Wrappers (LDWs) (S08, S12, S13) entails consideration of various technologies, including the Resource Description Framework (RDF) and the RDF Query Language (SPARQL). Combining Linked Open Data (LOD) with ontologies establishes a cloud-based structure based on an ontology-driven open data architecture. This architecture serves as an intermediary between OD resources and data-mining tools. Furthermore, when integrating data hosted by different tools, LOD considers that OD is linked and accessed using specific technologies, such as RDF.

We observe the use of "uniform data structures and open architectures" and "open data clouds" regarding **Open Platforms**. Prioritizing standardization and readability is essential to achieve uniform data structures and open architectures (S06). Regarding open data clouds (S08), a middleware-based architecture is utilized to leverage open data. The middleware is structured into layers related to Data Coordination, Processing, and the API. The coordination layer handles the sending and receiving of data from a cloud-based system for open data, using LOD and ontologies. In the processing layer, various calculations and operations are carried out. The API layer uses the middleware.

In the context of **Open Review**, the "exemplar process guideline" (S09) aims to illustrate important aspects to consider during a review. It covers application, specific attributes, general quality criteria, anti-standards, acceptable deviations, and invalid and exemplary criticisms.

Open Access (S10) is related to the practice of a *framework for analyzing open access journals*, built on a set of attributes (bibliographic information, activity metrics, economics, accessibility, and predatory issues). Finally, observations are presented to readers of open-access journals.

Artifact Sharing (S11) is considered by the practice *a plan for creating and sharing artifacts*. The plan must consider different perspectives of the creators, mentors, reviewers, and organizers. For each perspective, observations were presented.

When leveraging **Open-Source** Software (S14) communities, the following actions can be taken to take advantage of their benefits: identify key software project repositories;

search repositories for software projects based on specific attributes; select five open source projects similar in size, scope, and complexity; review the developer and discussion forums related to each of the five projects to determine experiences related to time, cost, and scope.

4.5.3 RQ2 - Challenges Related to Open Science in Software Engineering

Table 8 outlines the observed challenges.

Table 8. Observed Challenges.

OS Challenges	Challenge	Study ID	Challenge ID
Open Data	data set update	S01	Ch.01
	enable the programming at an end-user level for inexperienced developers; minimize maintenance efforts	S02	Ch.02
	deal with licensing, governance, and privacy	S05	Ch.03
	tools for dealing with open data	S12	Ch.04
	linked data wrapper maintenance	S13	Ch.05
Open Access	deal with data security (separation for private and public access)	S03	Ch.06
	preconceptions related to open access	S10	Ch.07
Artifact Sharing	changing of a cultural paradigm, related to data sharing	S06	Ch.08
	deal with multiple and decentralized data sources; lack standards for data publishing	S07	Ch.09
	deal with portability, maintenance and hosting questions; lack of standards and guidelines	S11	Ch.10

Regarding **Open Data**, the challenge of dataset updates (S01) involves managing multiple datasets within a software portal. This may be complicated by different management strategies, such as organizing multiple dataset versions into a single package using the Comprehensive Knowledge Archive Network (CKAN) or managing datasets individually with Socrata.

When practitioners consider enabling programming at the end-user level (S02), they must assess the potential for users to use applications in unforeseen ways.

In addition to security, it is also essential to address licensing, governance, and privacy (S05). Licensing primarily concerns the use of open-source tools. Governance and privacy are closely linked to data sharing, even in the context of OD in software development (e.g., static code analyzers). In S05, the authors discuss the challenges of sharing development data and balancing the privacy of business information.

In S12, the authors emphasize the importance of tools for handling OD. The study demonstrates the combined use of the tools Abreforjas (S04), EasyData/Rails, EasyData/Django, and the Data adapter for publishing data from Enterprise Architect.

To support the maintenance of linked data wrappers (LDWs) (S13), it is recommended that, when publishing an LDW, a maintenance strategy be established to ensure consumers of the wrapper's evolution. The authors emphasize the importance of adaptive maintenance for new environments. It is crucial to ensure the evolution of APIs and Ontologies and update LOD to meet new requirements in new environments.

Regarding **Open Access**, in data security (separation of private and public access) (S03), different approaches can be observed within a layer-based architectural model. One is to remove or move one or more layers to an intranet to limit that layer's audience. Another approach is to determine access to different layers based on users' needs through an access control mechanism.

In the context of cultural paradigms, it is important to address preconceptions related to open access (S10). These biases can be minimized by improving the transparency and quality of presentation on journal websites and open-access services.

For **Artifact Sharing** it is important to focus on efforts that contribute to changing the cultural paradigm related to data sharing to address the challenges observed in the context of open source in SE. In S06, the authors present evidence suggesting that data sharing is more of a cultural problem than a technical problem. In this context, it is essential to raise awareness that such data can be valuable to various user groups, including developers and non-developers.

Given the challenge of managing multiple decentralized data sources and the lack of standards for data publication, database integration is likely inadequate, despite the need. For example, in S07, the authors report inadequate integration across databases. Regarding standardization, the authors note the absence of standards, particularly for data sources and formats. Without standards, diverse situations emerge. For instance, data on the same hospitals may be published as Web Services in Rio de Janeiro and as Comma-Separated Values (CSV) in Recife.

Other important considerations include addressing portability, maintenance, and hosting issues, as well as the lack of established standards and guidelines (S11). Portability issues may arise when an artifact cannot run on other computers because of incomplete experimental packages, unresolved dependencies, or source code issues. Maintenance can become challenging when the original creator is no longer involved. Hosting problems may arise when storing large artifacts becomes challenging, necessitating high-storage-capacity repositories. The absence of standards and guidelines is evident when creators and reviewers are uncertain about how artifacts should be packaged and which standards to follow.

4.5.4 RQ3 - Topics Related to Open Science

In addition to practices and challenges, it is important to understand the OS topics addressed in the selected studies.

Table 9 provides specific details for each study topic.

Table 9. Open Science Topics.

OS Topics	Studies
Open Data	S01; S02; S04; S06; S12
Open Government	S03; S07; S08
Open Source Software	S05; S14
Open Review	S09
Open Access	S10
Open Reproducible Research	S11
Open Platform	S13
Open Feedback	S15

Analyzing the studies in Table 5, we observed that they were directly related to at least one branch in the OS Taxonomy.

In the context of **Open Data**, various structures are associated with the findings presented in the studies. Exam-

ples include platforms, frameworks, and mashups. In S01, open data platforms such as CKAN, Socrata, ArcGIS Open Data, and OpenDataSoft are observed. The study presents a methodology for treating open data portals as software platforms. In S06, practices are suggested to promote the openness of software platforms, such as Integrated Development Environments (IDEs) and Virtual Machines (VMs), by increasing the availability of shared data and metadata.

In S04, an open data framework is proposed as part of an assessment approach that focuses on learning in collaborative software development projects. The framework involves working with data sources and representation models, taking assessment criteria into account. Furthermore, the framework described in S12 facilitates the creation of automated evaluation processes at the outset of each software project. This framework is explained through its activities and models.

The mashup concept is introduced in S02 for use in software ecosystems that rely on open data. An ecosystem based on mashups involves users, authors, and service providers as key actors. The implementation of a mashup application is described in terms of engines, clients, and technology characteristics.

In the context of **Open Government**, our selected studies offer solutions for architecture and repositories. In S03, an architectural model for designing software platforms is presented, focusing on publishing open data, information, and knowledge. This model defines various operations. S08 introduces a middleware-based architecture. This architecture is an intermediary between open data resources and data mining tools. The middleware can be used through requests and responses.

S07 describes two Brazilian initiatives, detailing the creation of repositories to store information about the cities of Rio de Janeiro and Recife. The study discusses the results achieved, challenges, and difficulties, emphasizing the potential for developing applications from the repository.

Regarding **Open Source**, in S05, the authors studied the use and sharing of static code analyzers and associated data. The results highlighted the need to develop solutions for sharing data, mitigating risks, and facilitating the use of tools such as analyzers. Additionally, S14 presented an evaluation of practices that can be integrated into SE studies to enhance student learning. They also introduced an approach to planning SE projects that is influenced by open-source software.

Regarding **Open Review**, S09 presents a survey of review approaches. The survey focused on reviewers of SE studies. We discuss issues concerning the acceptance, conduct, and reporting of review results.

Considering **Open Access**, S10 presents a framework for analyzing open-access journals in SE and information systems. The framework was developed based on attributes such as bibliographic information, accessibility, and predatory issues. Additionally, the study offers valuable observations.

Regarding **Open Reproducible Research**, one crucial aspect of OS is artifact management, which promotes open and reproducible research. S11 discusses a study examining the challenges of creating, sharing, and using artifacts. The study's methodology involved collecting and analyzing primary and secondary data, and also provided insights for creators, advisors, reviewers, and event organizers.

Considering **Open Platforms**, they can contribute to various aspects of the OS, such as open data and open research. In S13, we constructed an LDW-based platform. We used an experimental framework to evaluate the platform.

Regarding **Open Feedback**, in S15, we developed a data-driven development model based on open feedback. The model was developed following a case study that identified several issues, including developing features without confirmed customer value, unclear feature content, and misalignment between the product and customer needs.

4.5.5 RQ4 - Contexts Related to Open Science and Software Engineering

Table 10 lists studies related to the contexts observed, including research, experience reports, development, and education.

Table 10. Contexts related to the Selected Studies.

OS Contexts	Studies
Research	S01; S02; S09; S10; S11
Experience Report	S03; S04; S06
Development	S05; S07; S08; S12; S13; S15
Education	S14

Five studies addressed the **research** context. A methodology was developed in S01 to investigate open data platforms. It involves searching for data portals in a specific area, developing a repository of data portals, identifying platforms, and collecting metadata. In S02, mashups are investigated, and the implementation of a mashup application is presented. S09 describes a survey conducted with reviewers of studies in SE. The survey aimed to document review practices. S10 presents a framework in an open-access context. Additionally, S11 addresses artifact management, considering the involvement of various actors who handle and oversee artifacts, including creators, advisors, and reviewers.

Three studies considered **experience reports**. S03 presents an architectural model for software platforms that considers open data publishing. The model establishes levels associated with open data, information, and knowledge and defines a set of operations for each level. Additionally, the study describes three application instances after defining the model. S04 outlines a learning-assessment methodology and an open-data framework for collaborative software development projects. It includes an analysis of evaluation examples and a case study. Additionally, S06 presents two experience reports describing projects that enhance the openness of software platforms, such as IDEs and VMs.

Six studies focused on the **development** context. In S05, the author investigates the use and sharing of static code analyzers across various scenarios, including program analyzers and the data they generate. Another study, S07, describes two Brazilian initiatives related to OD repositories, which record city characteristics. These repositories can utilize data from various sources, and applications can extract and present different information to citizens and tourists in these cities. S08 introduces a middleware-based architecture that incorporates ontologies and LOD. The architecture

provides an API for use. Another study, S12, discusses developing a framework for automated evaluation of software projects. The framework's evaluation is based on a case study. S13 presents the development of Linked Data wrappers (LDWs) and assesses their maintenance. S15 examines data-driven development, which involves developing a model to guide development using a case study.

S14 addressed the **education** context by evaluating practices that can be integrated into SE studies to promote student learning. It presented an approach to project planning.

4.5.6 RQ5 - Evidence Types and Empirical Robustness

To answer RQ5, we examined how each primary study empirically supports its Open Science-related practices and propositions. For each study, we identified the primary empirical strategy adopted, the scope of the evidence provided, and the overall level of robustness. The classification considers the clarity of the research design, the volume and relevance of the data, and the explicitness of the analytical procedures.

Table 11 presents a detailed overview of the empirical strategies and robustness assessments for each study. The evidence base is heterogeneous. Only a subset of studies provides systematic and relatively strong empirical support. In contrast, many others rely on design-oriented contributions, experience-based accounts, or prototype evaluations that offer limited or context-specific validation.

Table 11. Empirical evidence types and robustness for the selected studies (RQ5).

Study	Main OS related focus	Empirical strategy	Robustness assessment
S01	Open Data platforms and portals	Large-scale automated observational study of 3,152 portals and 1.9M datasets.	High robustness, supported by a large and diverse sample and a well-described methodology.
S02	Mashup-based Open Data ecosystems	Prototype with usage scenarios.	Low to moderate robustness, based on a single proof of concept with mainly descriptive evaluation.
S03	Layered architecture for Open Data and Open Government	Architecture instantiated in a real platform.	Moderate robustness, grounded in concrete examples but without systematic quantitative assessment.
S04	Open Data framework for sustainable assessment in software forges	Case study in a learning environment.	Moderate robustness, supported by concrete metrics but limited to a single educational setting.
S05	Open Data-driven improvements of static code analysis	An exploratory qualitative study with focus groups.	Moderate robustness, with a clearly described method but a small and context-specific sample.
S06	Open platforms and Open Data in IDEs and VMs	Experience report.	Low robustness, based on practitioner insights without formal data collection.
S07	Brazilian government Open Data applications	Experience report.	Low robustness, reflecting development experiences without systematic measurement.
S08	Middleware architecture for Business Intelligence	Prototype and scenario-based illustration.	Low robustness, mainly conceptual, with no structured user evaluation.
S09	Open review practices in Software Engineering	Mixed methods study with interviews and a survey of 175 reviewers.	High robustness, due to methodological clarity and substantial sample size.
S10	Open Access journals in SE and IS	Structured content analysis of 30 journals.	High robustness, supported by systematic sampling and a clearly defined analytical framework.
S11	Artifact sharing and Open Reproducible Research	Mixed methods study with two complementary surveys.	High robustness, with triangulated data sources and large participant samples.
S12	Framework for software process deployment and evaluation	Instantiated framework in an industrial-style scenario.	Moderate robustness, based on a realistic use case but without broader comparison.
S13	Linked Data Wrapper platform maintenance	Prototype with a user study involving 12 participants.	Moderate robustness, with explicit evaluation, although based on a small sample.
S14	Data-driven planning in SE curricula	Exploratory educational and documentary analysis.	Low robustness, mainly conceptual and reflective in nature.
S15	Data driven software R&D and feedback models	Longitudinal multi-case study in three companies.	High robustness, supported by detailed multi-organizational and long-term evidence.

The empirical strategies observed across these studies (Table 11) illustrate the diversity of methodological choices in Open Science research within Software Engineering. Table 12 synthesizes this distribution by grouping the studies according to their primary empirical approach. The table shows that only a few studies employ highly structured methodologies, such as large-scale observational analyses (S01), mixed-methods designs with substantial samples (S09 and S11), or longitudinal multi-case investigations (S15). These studies provide stronger empirical grounding for their claims.

A second group of studies follows structured but localized validation strategies. This group includes single case studies, instantiated frameworks applied in practice-based contexts, exploratory qualitative studies with defined protocols, and evaluated prototypes. Such studies (S03, S04, S05, S12, and S13) provide meaningful insights, but their empirical support remains context-dependent and limited in generalizability.

In contrast, many contributions rely on less formalized empirical strategies, such as prototypes illustrated only through examples (S02, S08, and S13), experience reports from development or educational settings (S06, S07, and S14), or conceptual reflections without systematic data collection. These studies enrich the understanding of emerging Open Science practices but provide weaker empirical support.

Table 12. Grouping of the 15 studies by empirical strategy (RQ5).

Empirical Strategy	Studies
Large-scale observational study	S01
Mixed methods study	S09, S11
Longitudinal or multi-case study	S15
Single case study or instantiated framework	S03, S04, S12
Prototype or implementation with illustrative evaluation	S02, S08, S13
Experience report or conceptual evidence	S06, S07, S14
Exploratory qualitative study	S05
Documentary or educational analysis	S14

To complement this view, Table 13 organizes the studies by robustness level. This perspective highlights how empirical rigor varies across strategies. Studies classified as highly robust consistently use structured research designs, substantial sample sizes, or long-term, multi-organizational evidence. Studies with moderate robustness employ structured yet localized approaches, whereas those with low robustness rely on descriptive accounts or conceptual discussions.

Table 13. Grouping of studies by robustness assessment and associated empirical strategies (RQ5).

Robustness Level	Empirical Strategies Present	Studies
High	Large scale observational, mixed methods, longitudinal multi case	S01, S09, S10, S11, S15
Moderate	Single case study, instantiated framework, exploratory qualitative, prototype with structured user study	S03, S04, S05, S12, S13
Low to moderate	Prototype with limited validation	S02
Low	Experience reports, conceptual evidence, prototype without structured evaluation	S06, S07, S08, S14

From a global perspective, five studies provide comparatively strong empirical support for their Open Science-related propositions. These studies rely on substantial samples, long-term observation, or explicit methodological triangulation. Another group of studies offers moderate robustness through structured but localized investigations. The remaining studies are exploratory or experience-driven, providing valuable insights but limited empirical depth.

In summary, the empirical foundation underlying Open Science practices in Software Engineering remains uneven. Although some studies adopt rigorous methodologies, many contributions validate their proposals only at a local scale or through illustrative evidence. Future research would benefit from stronger empirical strategies, such as replications,

cross-context analyses, and multi-organizational longitudinal designs, to better capture the effectiveness and feasibility of Open Science practices in the discipline.

4.5.7 RQ6 - Alignment with UNESCO Open Science Principles

RQ6 examines how the Open Science-related contributions identified in the selected studies align with the four overarching UNESCO Open Science principles: transparency, accessibility, reproducibility, and collaboration. This perspective provides an internationally recognized lens that complements the internal structure of the FOSTER taxonomy and enables a broader assessment of how Software Engineering research operationalizes Open Science in practice.

Overall, alignment with the UNESCO principles is uneven and typically partial. Transparency emerges as the most frequently supported principle. Several studies provide detailed methodological descriptions, explicit architectural specifications, or clearly documented data-processing workflows, making their processes understandable and open to scrutiny. For example, S01 describes its portal-discovery and metadata-extraction pipeline in detail, while S09 and S11 thoroughly document their survey instruments, sampling procedures, and analytical steps.

Accessibility is most evident in studies centered on open data infrastructures, platforms, or portals. S01 and S07 explicitly focus on publishing and exploring open data portals at governmental or public scales. At the same time, S03 proposes layered architectures that facilitate the dissemination and consumption of open data and open government information. However, accessibility is often narrowly framed in terms of data availability, with less emphasis on persistent identifiers, interoperable metadata, and long-term stewardship, which are central to UNESCO's formulation.

Reproducibility appears only sporadically. Few studies provide conditions that would support full replication of their analyses or reuse of their datasets. S01 is closest to strong alignment, offering a clearly described large-scale pipeline applied to a substantial corpus of portals and datasets. However, the long-term availability of extracted data and scripts is not fully discussed. S09 and S11 describe replicable survey workflows, but reproducibility would require releasing questionnaires, codebooks, and analysis scripts. Most design-oriented contributions and experience reports (e.g., S06, S07, S08, and S14) describe systems or processes without providing the artifacts or protocols necessary for rigorous reproduction.

Collaboration is the least systematically supported principle. It appears primarily in studies of ecosystems, shared infrastructure, or multi-stakeholder environments. S02 discusses mashup ecosystems involving end users, authors, and service providers. S03 and S07 describe open data initiatives that, in principle, allow external developers to build new applications. S15 reports multi-company experimentation requiring collaboration between product teams and researchers. Even in these cases, however, mechanisms for participatory governance, co-creation, or community feedback loops are typically conceptual rather than empirically evaluated.

Table 14 summarizes the alignment of each study with the four UNESCO principles, using a three-level scale (low, moderate, strong) derived from internationally recognized conceptualizations of Open Science [Pontika *et al*., 2015; UNESCO, 2022]. A study was coded as **strongly aligned** when it operationalized a principle through concrete artifacts, openly accessible datasets, explicit methodological protocols, or platform mechanisms that demonstrably support transparency, accessibility, reproducibility, or collaboration. A **moderate alignment** indicates conceptual or partial support, such as descriptive workflows or semi-open resources that encourage openness but do not fully enable replication or structured participation. A **low alignment** reflects minimal or incidental engagement, typically limited to narrative statements without providing artifacts, access mechanisms, or procedural details.

Table 14. Alignment of studies with UNESCO Open Science principles (RQ6).

Study	Transparency	Accessibility	Reproducibility	Collaboration
S01	Strong	Strong	Moderate to strong	Moderate
S02	Moderate	Moderate	Low	Moderate
S03	Moderate	Strong	Low	Moderate
S04	Moderate	Low	Low	Low
S05	Moderate	Low	Low	Low
S06	Low	Low	Low	Low
S07	Moderate	Strong	Low	Low
S08	Low	Moderate	Low	Low
S09	Strong	Low	Moderate	Low
S10	Strong	Moderate	Moderate	Low
S11	Strong	Moderate	Moderate	Low
S12	Moderate	Low	Low	Low
S13	Moderate	Low	Low	Moderate
S14	Low	Low	Low	Low
S15	Moderate	Moderate	Moderate	Moderate
Count Strong	4	3	0	0
Count Moderate	8	5	4	5
Count Low	3	7	10	10
Count Moderate-Strong	0	0	1	0
TOTAL	**15**	**15**	**15**	**15**

Taken together, these findings show that Software Engineering research has begun to incorporate Open Science practices, but typically in a fragmented manner when viewed through the UNESCO principles. Established habits regarding methodological clarity and the availability of data or tools ensure that transparency and accessibility are addressed more consistently. In contrast, reproducibility and collaboration remain underdeveloped, often lacking concrete artifact packages, shared protocols, or empirically evaluated mechanisms for co-creation and community engagement. As the field advances, closer alignment with UNESCO Open Science principles will require not only improved technical infrastructures but also more substantial cultural and institutional incentives for sharing, reuse, and collaborative governance of research artifacts and processes.

4.5.8 RQ7 - Reproducibility Support in Terms of Availability, Completeness, Traceability, and Verifiability

RQ7 investigates the extent to which the selected studies enable reproducibility, as assessed using four criteria adapted from ACM Artifact Review and Badging guidelines and artifact evaluation practices at ICSE and ESEM: availability, completeness, traceability, and verifiability. In this context, *availability* refers to whether data, code, or other research artifacts are made accessible to readers in a stable, documented manner. *Completeness* captures whether the provided artifacts form a coherent package that includes all essential components needed to re-run analyses or reproduce

results. *Traceability* concerns the explicit linkage between elements in the paper (e.g., figures, tables, research questions) and their corresponding artifacts, datasets, or scripts. Finally, *verifiability* reflects whether a third party would be able to follow the documented procedures and artifacts to independently check or recompute at least part of the reported results.

Overall, the analysis indicates that explicit and robust support for reproducibility is limited across the fifteen studies. Most contributions do not provide complete artifact packages as defined in ACM artifact evaluation tracks and therefore do not qualify for the stronger badges associated with replicable or reusable research. Instead, most studies provide at best partial support, in the form of methodological descriptions, informal pointers to datasets, or high-level descriptions of tools and platforms.

In terms of availability, very few studies explicitly commit to public repositories or stable artifact hosting. S01 and S07 rely on open data portals and thus indirectly benefit from publicly accessible datasets, but they do not clearly state that they release processed data and scripts. Studies such as S09, S10, and S11 provide relatively rich methodological descriptions and sometimes reference online materials. Yet evidence for systematic artifact packages that include code, analysis scripts, or archival datasets remains limited. The remaining studies mostly describe tools, architectures, or experiences without making the associated artifacts accessible.

Completeness is even less supported. None of the studies clearly provides an integrated package that combines data, code, configuration, documentation, and execution instructions in a form that aligns with current best practices for artifact completeness in the ICSE and ESEM artifact tracks. At best, some studies provide only partial elements, such as detailed descriptions of sampling and coding procedures, without the corresponding raw data, scripts, or configuration files. Design-oriented or experience-based contributions (e.g., S02, S06, S07, S08, S14) typically remain at the level of conceptual or architectural exposition.

Traceability is somewhat better addressed in studies with stronger empirical designs. The studies S01, S09, S10, S11, and S15 establish reasonably clear links between research questions, data sources, and analysis procedures. Even so, these links are documented primarily in the text rather than through explicit mechanisms that link artifacts to paper sections. For instance, persistent identifiers or fine-grained cross-references that map tables and figures to specific datasets or scripts are rare. In studies focused on frameworks, architectures, or educational experiences (S03, S04, S12, S14), the traceability between conceptual elements and any underlying data or tools is primarily descriptive.

Verifiability is the most demanding criterion and therefore the least supported. Only a small subset of studies provides sufficient detail and structure to enable an independent researcher to conduct a partial reanalysis. S01 offers a clearly described pipeline that, in principle, could be reimplemented given sufficient effort and access to the same portals. S09 and S11 provide replicable methodological workflows for surveys and artifact analyses, but the absence of publicly available instruments and data limits the verifiability of these methods. S15 describes its longitudinal, multi-case design in a way that supports analytical understanding; however, the industrial nature of the data and the lack of open-source packages limit direct verification. In most other studies, the absence of explicit artifacts, scripts, or execution instructions renders verifiability very low.

Table 15 synthesizes the qualitative assessment of the four criteria for each study using a three-level scale (low, moderate, high). The classification reflects how the reported practices, descriptions, and references would support a reproducibility-oriented reading of the work, rather than a formal artifact evaluation in the sense of ACM badging.

Table 15. Qualitative assessment of reproducibility support across the selected studies (RQ7).

Study	Availability	Completeness	Traceability	Verifiability
S01	Moderate	Moderate	Moderate to high	Moderate
S02	Low	Low	Low	Low
S03	Low	Low	Low	Low
S04	Low	Low	Moderate	Low
S05	Low to moderate	Low	Moderate	Low
S06	Low	Low	Low	Low
S07	Moderate	Low to moderate	Low	Low
S08	Low	Low	Low	Low
S09	Moderate	Moderate	Moderate	Moderate
S10	Moderate	Low to moderate	Moderate	Low to moderate
S11	Moderate	Moderate	Moderate	Moderate
S12	Low	Low	Low to moderate	Low
S13	Low	Low	Low	Low
S14	Low	Low	Low	Low
S15	Low	Low	Moderate	Low
Count Moderate	5	3	6	3
Count Low	9	10	7	11
Count Low-Moderate	1	2	1	1
Count Moderate-High	0	0	1	0
TOTAL	**15**	**15**	**15**	**15**

Taken together, these results suggest that, while several studies pay careful attention to methodological clarity and conceptual rigor, they rarely translate these efforts into concrete artifact packages that meet contemporary expectations for reproducibility in Software Engineering. Very few contributions would qualify for higher-level ACM artifact badges, such as the replicated or reusable badges. Most would, at best, provide only partial availability or documentation, and some would not meet even those criteria. This indicates a substantial opportunity for future work on Open Science in Software Engineering to move beyond conceptual and infrastructural discussions and toward systematically curated, complete, and verifiable artifact sets that support independent reproduction and reuse of results.

4.5.9 RQ8 - Research Gaps, Inconsistencies, and Opportunities

RQ8 investigates the research gaps, inconsistencies, and opportunities that emerge when the 15 selected studies are analyzed collectively, aiming to outline a structured research agenda for Open Science in Software Engineering. The previous research questions showed that Open Science-related work in SE is still fragmented, with an emphasis on specific dimensions such as Open Data, portals, and artifact sharing. At the same time, other principles and practices remain underexplored. By synthesizing the evidence from S01–S15 across practices (RQ1), challenges (RQ2), topics (RQ3), contexts (RQ4), empirical robustness (RQ5), alignment with UNESCO principles (RQ6), and reproducibility criteria (RQ7), it is possible to identify coherent lines for future research.

From a cross-cutting perspective, several gaps and inconsistencies stand out. First, there is a clear imbalance be-

tween transparency and accessibility, on the one hand, and reproducibility and collaboration, on the other. Many studies describe architectures, platforms, or portals that increase data availability (e.g., S01, S03, S07), but only a few provide concrete conditions for reproducible research or structured community participation (e.g., S01, S09, S11, S15). Second, empirical support is uneven. A small subset of works relies on robust designs such as large-scale observational analyses, mixed methods studies, or longitudinal industrial cases (for example, S01, S09, S10, S11, S15), while many contributions are design-oriented, experience-based, or conceptually driven with limited validation (for example, S02, S06, S07, S08, S14).

Third, evidence linking Open Science practices to measurable effects on software engineering outcomes remains limited. Studies often motivate Open Data, open platforms, or artifact sharing in terms of expected benefits such as improved quality, better decision-making, or enhanced learning, but rarely provide systematic evidence that quantifies or traces these effects over time. Fourth, theoretical foundations from Open Science and related fields are only partially integrated. While FOSTER and related taxonomies are used as classificatory devices, there is little engagement with broader theories of openness, governance, incentives, or socio-technical change to explain why OS practices succeed or fail in SE contexts.

Fifth, the existing work is strongly tool- and infrastructure-centric, prioritizing portals, middleware, frameworks, or platforms (e.g., S01-S04, S06-S08, S12, S13). Organizational, cultural, and institutional dimensions associated with Open Science adoption are largely treated as informal observations or challenges rather than as primary objects of empirical investigation. Sixth, the educational and capacity-building perspective remains underdeveloped. S14 is an important starting point, but little systematic knowledge remains about how to train students, practitioners, and researchers to apply OS practices sustainably in SE. Finally, few replications or comparative studies exist across different communities, domains, or regions. Most evidence is local, anchored in specific projects, organizations, or national initiatives.

To consolidate the cross-study analysis performed for RQ8, Table 16 summarizes the main research gaps identified across the fifteen selected studies, the specific evidence that illustrates each gap, and the corresponding opportunities for advancing Open Science in Software Engineering. By mapping each gap to the research agenda items proposed in this section, the table highlights how current limitations in reproducibility, empirical robustness, socio-technical understanding, educational integration, and Open Science governance can be systematically transformed into actionable directions for future work. This structured synthesis clarifies where the existing body of knowledge is fragmented or underdeveloped and provides a coherent bridge from observed issues to targeted research opportunities.

Based on these observations, we propose a structured research agenda for Open Science in Software Engineering that builds on the existing contributions but moves beyond their current limitations:

Table 16. Synthesis of research gaps, evidence, opportunities, and agenda items (RQ8).

Identified Gap	Studies Illustrating the Gap	Research Opportunity	Related Agenda Item
Strong emphasis on transparency and accessibility, weak on collaboration and reproducibility	S01, S03, S07, S09, S10, S11, S15	Design multidimensional OS evaluation frameworks covering all UNESCO principles	RA1
Limited support for reproducibility and lack of artifact packages	S02, S03, S04, S06, S07, S08, S12, S13, S14	Develop complete, reusable, openly available artifact sets following ACM/ICSE guidelines	RA2
Local, single-context validations with limited generalizability	S02, S03, S04, S05, S06, S07, S08, S12, S13, S14	Conduct longitudinal and multi-organizational studies on OS adoption dynamics	RA3
Predominance of tool- and platform-centric approaches over socio-technical analysis	S01, S02, S03, S04, S06, S05, S08, S12, S13	Investigate OS adoption using socio-technical, organizational, and governance theories	RA4
Sparse integration of educational OS practices in SE curricula	S14 (mainly), S02, S07	Develop and evaluate OS-oriented educational interventions and competency models	RA5
Lack of metrics, indicators, and maturity models for OS in SE	Across all studies, implicit in S01, S03, S10, S11, S15	Define measurable indicators and maturity models for OS implementation in projects and venues	RA6
Insufficient empirical evaluation of OS-enabling infrastructures and policies	S01, S02, S03, S04, S06, S07, S08, S12, S13	Co-design and empirically assess infrastructures, policies, and community guidelines for OS	RA7

- **RA1. Multi-dimensional evaluation of Open Science interventions in SE.** Future work should design and conduct studies that jointly assess transparency, accessibility, reproducibility, and collaboration, rather than focusing on single dimensions. Building on topics such as Open Data platforms and artifact sharing (for example, S01, S03, S07, S11), new research can propose evaluation frameworks that relate OS practices to technical, social, and organizational outcomes in software projects.
- **RA2. Replication and reproduction studies with complete artifact packages.** In line with ACM and ICSE/ESEM artifact guidelines, there is a need for studies that publish complete and reusable artifact sets, including data, code, scripts, and documentation, in order to support independent reproduction and replication. Existing work on artifact sharing and open, reproducible research (e.g., S11) can be extended through concrete replications of prior OS-related studies, including S01, S09, S10, and S15.
- **RA3. Longitudinal and multi-organizational studies of OS adoption.** The longitudinal industrial evidence in S15 illustrates the value of sustained observation. Future research should systematically investigate how OS practices are introduced, adapted, and institutionalized across multiple organizations and over longer periods. Such studies can examine how policies, incentives, and tooling interact with everyday development practices and decision processes.
- **RA4. Socio-technical and organizational change perspectives.** Many challenges reported in the selected studies, such as cultural resistance to data sharing, lack of standards, and concerns about privacy or licensing, are socio-technical in nature. Research should draw on theories and methods from organizational studies, sociology, and information systems to understand how Open Science practices can be embedded in existing SE processes, governance structures, and communities of practice.
- **RA5. Education, training, and competency building in OS for SE.** The educational work in S14 suggests that

curricula can incorporate data-driven and open practices, but the effects on skills, attitudes, and long-term behavior remain largely unexplored. Future studies should design and evaluate educational interventions that explicitly target OS competences in SE, including data management, artifact curation, reproducibility practices, and ethical and legal aspects of openness.

- **RA6. Metrics, indicators, and maturity models for OS in SE.** There is a need for conceptual and empirical work on how to measure the degree of openness of SE research and practice. Building on taxonomies such as FOSTER and the UNESCO principles, new studies can propose indicators, maturity models, and assessment instruments that capture how projects, research groups, venues, or organizations implement OS practices and how these levels evolve over time.
- **RA7. Design and evaluation of OS-enabling infrastructure and policies.** Several studies propose platforms, portals, and frameworks for Open Data and artifact management, yet systematic assessment of their adoption and impact is scarce. Future research should investigate how technical infrastructures, institutional policies, and community guidelines can be co-designed and assessed to effectively support OS in SE, including integration with venues that already implement badge systems or artifact tracks.

Taken together, these research directions indicate that Open Science in Software Engineering has progressed beyond conceptual discussion and isolated initiatives, but still lacks a coherent, empirically grounded body of knowledge. Addressing the identified gaps requires moving from tool-centric, locally validated proposals to multidimensional, reproducible, and theory-informed investigations that systematically connect OS practices with their technical, organizational, and societal effects in SE.

5 Discussion

This section discusses the findings presented in Section 4.5 by interpreting the evidence obtained for each research question and relating the identified patterns to broader Open Science (OS) challenges in Software Engineering (SE). Rather than repeating the descriptive results, this section synthesizes the evidence, examines its implications, and derives observations to support researchers and practitioners in adopting and advancing Open Science practices in Software Engineering. Each observation in this section is grounded in empirical evidence from the systematic review and highlights opportunities for future research, methodological improvements, or practical recommendations.

5.1 Discussion and Observations Related to Practices

The findings associated with **RQ1** show that Open Data is the most frequently reported Open Science practice among the selected primary studies. Most contributions emphasize the publication, organization, integration, and management of research data through repositories, open platforms, Linked Open Data (LOD), and data-driven software development approaches. This result indicates that, within the current Software Engineering literature, Open Science is predominantly interpreted from a data-centric perspective.

Although the reviewed studies demonstrate considerable maturity in data publication and organization, they pay considerably less attention to aspects such as long-term stewardship, governance, sustainability, provenance, interoperability, and systematic reuse. Consequently, the literature still presents a fragmented view of the complete data lifecycle, often emphasizing data availability while offering limited support for its ongoing management and reuse across software projects.

Several studies also demonstrate that integrating heterogeneous data sources remains challenging. While approaches based on Linked Open Data and mashups have shown promising results, their adoption remains limited, and their applicability across different Software Engineering contexts is still insufficiently investigated. This finding suggests that future research should focus not only on publishing data but also on facilitating its integration, interoperability, and long-term reuse.

> **Obs.01:** *Different open data sources from various software projects should be combined using the Mashup concept.*

The results further indicate that interoperability depends strongly on adopting common data standards and well-defined metadata structures. Several primary studies rely on structured representations to facilitate information exchange; however, no common standard has emerged across the analyzed literature. This lack of standardization increases integration effort and limits data reuse across projects and repositories.

> **Obs.02:** *Data standards and data life cycle models should be analyzed to ensure adherence to different data sets and facilitate data curation.*

Another recurring finding concerns the importance of Open Data Models. Although they provide a valuable foundation for organizing research artifacts, the reviewed studies rarely address complementary activities required throughout the data lifecycle, such as anonymization, redaction, provenance recording, preservation, and long-term curation. Therefore, current models only partially support the broader objectives advocated by Open Science.

> **Obs.03:** *Open Data Models are crucial for organizing data. To simplify curation activities, they should be thoroughly researched and implemented in Software Engineering research data.*

The evidence also indicates that effective data reuse depends not only on making datasets publicly available but also on documenting their provenance, evolution, and maintenance history. Without this contextual information, data sharing alone provides limited support for reproducibility and secondary reuse.

> **Obs.04:** *When defining and using open data, whether collaboratively or not, it is essential to consider models or standards that promote data reuse, thereby emphasizing data provenance. Reuse is a crucial aspect when creating and assessing open data models.*

Beyond data management, some studies show that software architectures themselves can facilitate openness by incorporating architectural abstractions that promote interoperability, extensibility, and information sharing. Considering these aspects during software design may reduce future barriers to adopting Open Science practices.

> **Obs.05:** *During software projects, abstractions that may favor the openness of software systems can be considered, such as open platform architectural models.*

Another important finding concerns the central role of repositories and specialized portals in preserving and disseminating research artifacts. The reviewed studies consistently recognize repositories as fundamental infrastructure for supporting long-term accessibility, version control, and artifact sharing.

> **Obs.06:** *Open data should be deposited in portals and repositories to facilitate their sharing and preservation, particularly in data-driven development.*

The analysis further reveals that interoperability becomes increasingly important when research artifacts are distributed across multiple repositories, organizations, or geographical locations. Provenance mechanisms, persistent identifiers, and interoperable metadata appear as essential enablers for connecting these heterogeneous environments.

> **Obs.07:** *Establishing interoperability between data or data sets should be prioritized, especially when they are stored in different or geographically distant repositories. This can be facilitated by implementing procedures related to data provenance.*

The reviewed studies also suggest that planning artifact management from the beginning of a research project contributes to better organization, preservation, and sharing outcomes. Rather than treating openness as a post-publication activity, Open Science practices should be incorporated into the research planning process.

> **Obs.08:** *The plan for creating and sharing artifacts and the long-term strategy for sharing and evaluating them should be established as a section of the research project.*

Finally, the evidence indicates that combining Linked Open Data technologies with mashup approaches represents a promising direction for improving integration among heterogeneous Software Engineering data sources. Although only a limited number of primary studies explicitly investigate this combination, the reported results suggest considerable potential for supporting richer research ecosystems.

> **Obs.09:** *The use of Linked Open Data and mashups might facilitate the integration of data from different sources, such as repositories or tools.*

5.2 Discussion and Observations Related to Challenges

The findings associated with **RQ2** indicate that the challenges reported in the primary studies extend well beyond technical limitations. Although issues related to data management and software infrastructure are frequently discussed, evidence shows that governance, maintenance, organizational culture, standardization, privacy, and security are equally important barriers to the effective adoption of Open Science practices in Software Engineering.

Among the identified challenges, long-term data maintenance appears as one of the least addressed aspects. While several studies focus on publishing datasets and making research artifacts publicly available, comparatively few discuss how these artifacts should be maintained, updated, versioned, or preserved throughout their lifecycle. This suggests that data sharing is often treated as the final stage of a project rather than as a continuous process requiring sustained management.

> **Obs.10:** *Establish data update procedures that should be clearly outlined in a research Data Management Plan (DMP).*

The reviewed studies further indicate that maintenance activities should be incorporated during the planning phase of research projects rather than being addressed only after data publication. Early planning facilitates resource allocation, documentation, preservation, and future reuse of research artifacts.

> **Obs.11:** *Consider data maintenance efforts from the early stages of research design. Actions related to these efforts should be clearly outlined in a Data Management Plan.*

Security also emerged as an important concern. Although some studies discuss secure data sharing and privacy preservation, systematic procedures for evaluating the security level of research datasets are rarely described. This finding suggests that security considerations remain largely implicit, potentially limiting confidence in openly shared research artifacts.

> **Obs.12:** *Include detailed assessment procedures for the security level of each data set in the Data Management Plan.*

The evidence further shows that licensing, governance, and privacy are frequently recognized as prerequisites for sustainable data sharing. However, these aspects are often discussed independently, and relatively few studies propose integrated governance approaches that balance openness with legal and ethical requirements.

> **Obs.13:** *Consider including licensing, governance, and privacy aspects in an Artifact Management Plan.*

Another recurring challenge concerns the lack of standardization for organizing, documenting, and sharing research artifacts. The diversity of formats, metadata descriptions, and management procedures reduces interoperability across repositories and complicates long-term reuse of Software Engineering research outputs.

> **Obs.14:** *Research artifact sharing standardization procedures and techniques should be explored to facilitate adoption and reuse.*

Overall, the evidence for RQ2 shows that many identified challenges stem from organizational, methodological, and cultural issues rather than purely technological limitations. Consequently, advancing Open Science in Software Engineering requires not only improved technical infrastructure but also governance mechanisms, institutional support, standardized practices, and long-term planning to sustain the entire research artifact lifecycle.

5.3 Discussion and Observations Related to Topics

The findings associated with **RQ3** reveal that Open Science research in Software Engineering is concentrated on a relatively small set of topics. As presented in Section 4.5, Open Data and Open Government constitute the dominant themes, whereas practices such as Open Reproducible Research, Open Protocols, Open Peer Review, Open Educational Resources, and Citizen Science receive considerably less attention.

This distribution suggests that current Open Science initiatives in Software Engineering remain largely focused on making research artifacts available, with comparatively less emphasis on practices that strengthen transparency, reproducibility, collaboration, and community participation throughout the research lifecycle. Consequently, the literature reflects a predominantly infrastructure-oriented view of Open Science rather than a holistic adoption of its principles.

The limited attention devoted to reproducibility-related topics is particularly noteworthy, as reproducibility is one of the fundamental pillars of Open Science. Likewise, the relatively small number of studies addressing collaborative mechanisms and open educational initiatives suggests opportunities to expand Open Science beyond data publication and repository use.

These findings indicate that future Software Engineering research would benefit from exploring a broader spectrum of Open Science practices, particularly those that improve research transparency, methodological rigor, collaborative knowledge production, and education.

> **Obs.15:** *Further explore how Open Science topics should benefit various research projects and artifacts, particularly Open Reproducible Research and protocols, as these topics are crucial to the field.*

5.4 Discussion and Observations Related to Contexts

The findings associated with **RQ4** show that Open Science practices have been adopted across different Software Engineering contexts, including academic research, software development, experience reports, and educational environments. However, their adoption remains uneven across these contexts.

The reviewed studies indicate that research-oriented contexts generally adopt more structured methodologies and provide richer methodological descriptions. In contrast, experience reports and development-oriented studies often offer valuable practical insights but tend to focus on localized solutions that are harder to generalize. Educational applications of Open Science appear less frequently in the analyzed literature despite their potential to promote long-term cultural change within the Software Engineering community.

These findings suggest that Open Science practices should be adapted according to the characteristics and objectives of each application context while preserving their fundamental principles of transparency, accessibility, reproducibility, and collaboration. At the same time, the limited number of educational initiatives highlights an opportunity to integrate Open Science concepts earlier into Software Engineering education and professional training.

> **Obs.16:** *Consult Open Science practices already used in the context of Software Engineering when planning the methodology of a study.*

5.5 Discussion and Observations Related to Evidence Types and Empirical Robustness

The findings associated with **RQ5** indicate considerable variability in the empirical evidence supporting Open Science research in Software Engineering. While some primary studies present rigorous empirical evaluations, including case studies, longitudinal analyses, and triangulated investigations, a substantial portion of the literature remains based on conceptual proposals, prototypes, or experience reports.

This variability suggests that the empirical foundations of Open Science research in Software Engineering are still evolving. Although conceptual contributions play an important role in introducing new ideas and infrastructures, stronger empirical evidence is necessary to assess their effectiveness, scalability, and applicability across different organizational and research settings.

Consequently, future research should place greater emphasis on robust empirical validation through comparative studies, replications, mixed-methods investigations, and longitudinal evaluations to provide stronger evidence regarding

the benefits and limitations of Open Science practices.

> **Obs.17:** *Strengthen empirical robustness by adopting structured, scalable, and triangulated methodologies when evaluating Open Science practices in Software Engineering.*

The reviewed studies also suggest that Software Engineering can benefit from empirical methodologies already consolidated in disciplines with a longer tradition of Open Science adoption, thereby accelerating the maturation of research practices within the field.

5.6 Discussion and Observations Related to Alignment with UNESCO Open Science Principles

The findings associated with **RQ6** indicate that the Open Science practices reported in the selected primary studies align only partially with UNESCO's principles. As discussed in Section 4.5, transparency and accessibility are the most consistently addressed dimensions, whereas reproducibility, collaboration, community participation, and long-term stewardship receive considerably less attention.

This imbalance suggests that Open Science adoption within Software Engineering remains largely focused on making research artifacts publicly available rather than supporting the broader collaborative ecosystem envisioned by UNESCO. Although open data and software constitute an essential step, sustainable Open Science also requires governance mechanisms, interoperable infrastructures, persistent identifiers, community engagement, and institutional support throughout the research lifecycle.

The reviewed studies further indicate that improving alignment with UNESCO principles requires coordinated advances in technical infrastructures, documentation practices, governance models, and collaborative research environments. Consequently, future initiatives should seek to balance technological solutions with organizational and socio-technical mechanisms that can sustain long-term Open Science ecosystems.

> **Obs.18:** *Promote broader and deeper alignment with UNESCO Open Science principles by strengthening reproducibility practices and integrating structured collaboration mechanisms into Software Engineering research workflows.*

> **Obs.19:** *Improve alignment with UNESCO accessibility principles by adopting persistent identifiers, interoperable metadata standards, and long-term preservation strategies.*

> **Obs.20:** *Increase transparency through richer documentation practices, including explicit disclosure of data flows, architectural decisions, and analytical assumptions.*

> **Obs.21:** *Develop collaboration-supportive infrastructures that embed participatory mechanisms directly into platforms and research tools.*

> **Obs.22:** *Encourage the use of community-driven governance models to enhance shared ownership and sustainability of Open Science infrastructures.*

5.7 Discussion and Observations Related to Reproducibility Support

The findings associated with **RQ7** reveal that reproducibility remains one of the least mature dimensions of Open Science in Software Engineering. Although many primary studies describe their methodologies and openly disseminate some research artifacts, relatively few provide complete, traceable, and reusable artifact packages that enable independent verification and reproduction of the reported results.

This finding exposes a gap between conceptual openness and practical reproducibility. Making artifacts publicly available does not necessarily guarantee that they can be understood, executed, reused, or independently validated. The reviewed studies indicate that missing documentation, incomplete datasets, lack of execution instructions, absent version information, and insufficient traceability remain common limitations.

Consequently, improving reproducibility requires a more comprehensive view of research artifact management, encompassing documentation, provenance, preservation, packaging, persistent identifiers, and independent verification throughout the research lifecycle.

> **Obs.23:** *Improve reproducibility support in Software Engineering research by providing complete, accessible, traceable, and verifiable artifact packages that enable independent reproduction and verification of results.*

Additional opportunities identified from the reviewed studies include:

> **Obs.24:** *Adopt standard artifact packaging templates that include data, scripts, documentation, and execution instructions.*

> **Obs.25:** *Use open repositories with stable versioning and persistent identifiers to ensure long-term accessibility.*

> **Obs.26:** *Introduce explicit traceability matrices linking claims, tables, figures, and datasets.*

> **Obs.27:** *Encourage independent verification exercises as part of the publication or artifact review process.*

Obs.28: *Incorporate reproducibility auditing into peer review to reinforce community norms and expectations.*

5.8 Discussion of Research Gaps and Future Opportunities

The synthesis of evidence across **RQ1–RQ7** reveals that Open Science adoption in Software Engineering has progressed considerably over the last decade but remains uneven across dimensions. While substantial efforts have focused on improving data sharing, repository infrastructures, and open platforms, comparatively fewer studies examine governance, interoperability, reproducibility, artifact lifecycle management, community participation, and long-term sustainability.

Taken together, the findings suggest that current research is still predominantly infrastructure-oriented. Most contributions concentrate on making research artifacts available, whereas fewer studies address how these artifacts should be documented, curated, maintained, evaluated, and reused throughout their lifecycle. Consequently, openness is often interpreted as publishing artifacts rather than as a continuous process that supports transparency, collaboration, and cumulative scientific knowledge.

Another important finding from the synthesis is the absence of widely adopted standards that uniformly describe Software Engineering research artifacts. Although several metadata models, ontologies, and repository solutions have been proposed, their adoption remains fragmented, reducing interoperability across repositories, projects, and research communities. This fragmentation also limits automated discovery, integration, reuse, and long-term preservation of research artifacts.

The review further indicates that reproducibility remains one of the least mature dimensions of Open Science within Software Engineering. Although datasets, source code, and other research artifacts are increasingly shared, relatively few studies provide sufficiently complete documentation, provenance information, execution environments, or validation procedures that enable independent reproduction of research findings.

Finally, the evidence highlights the importance of viewing Open Science as an integrated socio-technical ecosystem rather than as a collection of isolated practices. Technical infrastructures, metadata standards, governance models, documentation, institutional policies, and community engagement should evolve together to maximize transparency, reproducibility, collaboration, and long-term sustainability.

Based on the synthesis of the reviewed literature, the following opportunities deserve greater attention in future Software Engineering research.

Obs.29: *Develop comprehensive metadata models capable of representing the complete lifecycle of Software Engineering research artifacts, including provenance, preservation, reproducibility, and interoperability information.*

Obs.30: *Investigate standardized artifact management processes that integrate Data Management Plans, metadata standards, persistent identifiers, and preservation strategies throughout the research lifecycle.*

Obs.31: *Conduct more empirical evaluations of Open Science practices through longitudinal studies, industrial case studies, replications, and comparative investigations to strengthen the evidence base of the field.*

Obs.32: *Explore interoperability mechanisms capable of connecting heterogeneous repositories, software platforms, datasets, and research infrastructures using shared metadata and semantic technologies.*

Obs.33: *Promote stronger integration between Open Science practices and Software Engineering education to foster a culture of openness among future researchers and practitioners.*

Obs.34: *Investigate governance models that balance openness, privacy, security, ethical requirements, and legal constraints while supporting long-term sustainability of shared research artifacts.*

Obs.35: *Encourage the adoption of artifact evaluation, certification, and reproducibility assessment processes as complementary mechanisms for improving research quality.*

Obs.36: *Develop tool support that automates artifact documentation, metadata generation, provenance capture, and repository publication to reduce researchers' effort when adopting Open Science practices.*

Obs.37: *Promote integrated Open Science ecosystems in which repositories, metadata standards, persistent identifiers, governance mechanisms, and reproducibility infrastructures operate together to support the complete lifecycle of Software Engineering research artifacts.*

6 Threats to Validity

The threats associated with this review are discussed following the perspectives proposed by Ampatzoglou *et al*. [2019] and Verdecchia *et al*. [2023]. We consider three categories of validity: study selection, data, and research.

Study selection validity concerns whether the procedures used to identify and select primary studies adequately represent the relevant body of literature. To reduce risks during search strategy construction, we iteratively refined and validated the conceptual search string through multiple pilot searches in ACM Digital Library and IEEE Xplore, which provide expressive query mechanisms and are widely recognized in Software Engineering research. To maintain feasibility, the conceptual search strategy intentionally focused on the term "Software Engineering", which may have excluded studies adopting different terminology to describe related research. Moreover, although a single conceptual search strategy was defined, the selected digital libraries provide heterogeneous search capabilities, including differences in supported search fields, query syntax, and filtering mechanisms. Consequently, database-specific adaptations were necessary to execute semantically equivalent searches while preserving the original search intent. For example, some digital libraries support searches by title, abstract, and keyword, whereas others provide only title-based searches or use different query interfaces. These operational differences may have affected the recall of potentially relevant studies. To mitigate this threat, we searched eight complementary digital libraries, complemented the automated search with backward and forward snowballing following the recommendations of Wohlin [2014] and Badampudi *et al*. [2015], and made the complete search protocol publicly available in the replication package, including the search strategy, database-specific search strings, search fields, operational adaptations, and intermediate datasets.

Data validity concerns the accuracy, completeness, and representativeness of the extracted information. The relatively small number of primary studies increases the possibility that the review has not yet captured some Open Science practices. However, this reflects the emerging nature of Open Science research within Software Engineering rather than shortcomings in the review methodology. Since the intersection of Open Science and Software Engineering remains relatively unexplored, only a limited number of studies explicitly address it. To mitigate this threat, we conducted a preliminary literature review to refine the research questions and ensure they were supported by observable evidence. Publication bias remains a potential concern because the primary search relied on digital libraries. Therefore, we also performed backward and forward snowballing, which led to the identification of an additional primary study not retrieved during the automated search. Finally, to reduce subjectivity during data extraction and classification, we adopted the well-established FOSTER Open Science taxonomy as the primary analytical framework.

Research validity concerns the rigor, transparency, and reproducibility of the review process. To strengthen these aspects, we followed a review protocol grounded in established guidelines for systematic reviews and systematic mappings in Software Engineering [Khalil and Tricco, 2022; Munn *et al*., 2018; Petersen *et al*., 2015; Kitchenham *et al*., 2015]. Furthermore, we substantially expanded the online replication package by documenting the complete search, study selection, and data extraction protocols, along with the database-specific search strings, metadata, and intermediate datasets generated throughout the review. These materials enable independent researchers to inspect, reproduce, and critically assess every stage of the review process. Nevertheless, the novelty of investigating Open Science within Software Engineering limits direct comparisons with previous secondary studies, reducing opportunities for triangulation. Likewise, the diversity of Open Science practices and the heterogeneity of Software Engineering research contexts may limit the generalizability of some observations. To reduce these risks, researchers with expertise in Open Science, Software Engineering, and secondary study methodology evaluated the search strategy before execution.

Overall, although several mitigation strategies were employed, residual threats remain. These limitations primarily reflect the evolving state of Open Science research within Software Engineering and should be considered when interpreting the findings presented in this review.

7 Prospective Actions

Building on the results of this Scoping Review Study and the structured research agenda proposed in Section 4.5.9, this section outlines a set of prospective actions that the Software Engineering community can adopt to foster sustainable and effective Open Science (OS) practices. These actions translate the identified research gaps, empirical limitations, and opportunities into concrete steps for researchers, practitioners, educators, infrastructure providers, and policy-making bodies. They aim to operationalize the principles highlighted by UNESCO and address the challenges and inconsistencies identified across the reviewed studies.

7.1 Actions for Researchers

Publish complete and reusable artifact packages. Researchers should prepare and share reproducible artifact sets, including data, scripts, documentation, and execution instructions, in accordance with best practices from ACM and ICSE artifact evaluation processes. This directly addresses the reproducibility and verifiability gaps reported in the studies reviewed.

Adopt transparent and richly documented methodologies. Strengthen methodological transparency by explicitly reporting data flows, architectural decisions, assumptions, and intermediate processing steps. This will better align with UNESCO's transparency principle.

Conduct cross-organizational and longitudinal investigations. Since most existing evidence is local and context-dependent, researchers should design studies that track OS adoption across diverse settings and over time, as RA3 suggests.

Engage in replication studies. Replications of influential OS-related studies, including those involving large-scale observational data and mixed-methods research, are essential for strengthening empirical robustness.

7.2 Actions for Conferences and Journals

Promote artifact evaluation and badge systems. Venues should extend the use of artifact and reproducibility badges, following initiatives already adopted by EMSE, ICSE, ESEM, MSR, and SANER.

Encourage the submission of reproducible and open

studies. Calls for papers should explicitly outline expectations for data availability, methodological transparency, and the use of reproducibility packages.

Include Open Science criteria in review processes. Review forms can include OS-related questions that assess data availability, artifact completeness, and methodological openness.

7.3 Actions for Infrastructure and Tool Providers

Develop OS-enabling platforms with embedded collaboration mechanisms. Tools and portals should support participatory governance, user feedback, and community-driven curation to address the collaboration gaps identified in the reviewed studies.

Support persistent identifiers and interoperable metadata. Repositories should adopt DOIs, versioning systems, and metadata standards to strengthen accessibility and stewardship, in line with UNESCO's accessibility principle.

Co-design infrastructures with real SE communities. Infrastructure development should be driven by empirical evidence and user needs, reflecting the opportunities described in RA7.

7.4 Actions for Educators and Institutions

Integrate Open Science competencies into SE curricula. Educational programs should incorporate training in data management, reproducibility, ethical openness, and artifact curation to address the gaps identified in S14 and RA5.

Promote hands-on OS experiences. Students should engage in activities such as preparing reproducibility packages, participating in open peer review, or contributing to shared datasets.

Establish institutional policies supporting openness. Universities and research groups should develop policies that encourage early sharing, transparent documentation, and responsible data management.

7.5 Actions for Funding Agencies and Policy Makers

Require OS-compliant Data Management Plans. Funding bodies can mandate detailed DMPs specifying data sharing, preservation, documentation, and licensing strategies.

Support infrastructures and training for OS adoption. Funding calls should include dedicated resources for repository maintenance, tool development, and OS capacity building.

Promote evaluation frameworks aligned with international principles. Policies should encourage the use of multi-dimensional OS evaluation frameworks, as proposed in RA1, to ensure balanced consideration of transparency, accessibility, reproducibility, and collaboration.

7.6 Cross-cutting Actions

Beyond actor-specific responsibilities, the findings of this SRS indicate the need for actions that operate across institutional, technical, educational, and cultural boundaries. These cross-cutting actions address the structural limitations revealed in RQ5-RQ8 and help align the Software Engineering community with international Open Science principles.

Foster a culture of openness and shared responsibility. Sustainable OS adoption requires cultural change across researchers, reviewers, editors, institutions, and industry partners. Collective awareness initiatives, recognition mechanisms, and community-wide incentives can reduce resistance to data sharing, clarify expectations, and normalize transparent research practices.

Establish community standards for OS artifacts and metadata. The variability in artifact quality observed across studies points to the need for shared norms regarding formats, metadata schemas, documentation depth, and execution instructions. Researchers, venue organizers, and repository maintainers should co-develop such standards to enhance completeness, traceability, and verifiability.

Strengthen socio-technical integration in OS practices. Technical infrastructures alone are insufficient to support high-quality OS outcomes. Community governance, participatory processes, and joint stewardship models must be integrated into platform and policy design to address the socio-technical gaps highlighted in RQ8.

Develop shared indicators and maturity models for OS. The absence of metrics within the reviewed literature suggests the need for community-endorsed maturity models that capture openness levels at project, research group, and venue scales. Such indicators would enable progress monitoring, benchmarking, and evidence-based policy decisions.

Encourage multi-venue alignment of OS policies. Conferences, journals, funding agencies, and institutions should coordinate expectations for artifact availability, documentation, and reproducibility requirements. Harmonizing policies reduces ambiguity, improves compliance, and supports consistent community-wide adoption.

Invest in long-term stewardship and sustainability. Many OS-related challenges stem from the lack of durable infrastructures, persistent identifiers, and long-term hosting strategies. Collective investment in institutional repositories, national infrastructures, or discipline-specific hubs can ensure the longevity and accessibility of research outputs.

Promote interdisciplinary collaboration. Insights from information science, data governance, learning sciences, human-computer interaction, and organizational change can inform more holistic OS adoption. Interdisciplinary collaboration supports the development of theory-driven, empirically grounded approaches to OS implementation in Software Engineering.

Together, these cross-cutting actions complement the role-specific recommendations in this section and provide a structural foundation for strengthening Open Science practices across the entire Software Engineering ecosystem.

To complement the prospective actions outlined in this section, Table 17 summarizes cross-cutting actions spanning stakeholders and organizational layers. These actions address structural, cultural, and infrastructural conditions that influence the adoption of Open Science practices in Software Engineering. By consolidating shared needs such as standardization, sustainability, socio-technical integration, and coordinated policies, the table provides a unifying perspective that links individual actions to the broader systemic transformations required for effective, long-term OS implementation.

Table 17. Cross-cutting actions to strengthen Open Science adoption in Software Engineering.

Cross-cutting Action	Description	Expected Impact
Culture of openness and shared responsibility	Promote community-wide awareness, incentives, and recognition mechanisms to normalize data sharing, transparent reporting, and collaborative research behavior.	Strengthens community engagement, reduces cultural resistance, and fosters long-term OS adoption.
Community standards for artifacts and metadata	Establish common guidelines for formats, documentation, metadata schemas, versioning, and execution instructions.	Improves the completeness, traceability, and verifiability of artifacts across studies.
Socio-technical integration of OS practices	Incorporate governance, participatory processes, and shared stewardship into OS tools and platforms.	Enhances collaboration, sustainability, and alignment with UNESCO principles.
Shared OS indicators and maturity models	Define metrics and maturity levels for assessing OS implementation at the project, group, venue, and institutional levels.	Enables evaluation, benchmarking, and evidence-based policy-making.
Alignment of OS policies across venues	Coordinate expectations among conferences, journals, institutions, and funding agencies regarding data availability and artifact quality.	Reduces inconsistencies and increases compliance with OS requirements.
Long-term stewardship and sustainability	Invest in persistent repositories, stable hosting, identifier systems, and long-term data preservation mechanisms.	Ensures durable accessibility, responsible reuse, and reproducibility.
Interdisciplinary collaboration	Engage fields such as information science, governance, HCI, and organizational studies to inform holistic OS adoption.	Strengthens theoretical foundations and improves the socio-technical fit of OS solutions.

8 Final Remarks

This Scoping Review Study investigated how Open Science has been conceptualized, practiced, and supported within Software Engineering, revealing a landscape that is promising yet fragmented. By examining 15 primary studies across multiple Open Science dimensions, the review identified a diverse set of practices, including open data publishing, artifact sharing, platform development, methodological transparency, and efforts toward open and reproducible research. At the same time, the results show that such practices are unevenly distributed, often lack empirical grounding, and are rarely accompanied by complete artifact packages or structured mechanisms for collaboration and reuse.

The synthesis of findings across RQ1-RQ8 yielded observations that highlight both maturity gaps and opportunities to strengthen Open Science in SE. Transparency and accessibility are relatively well represented in the existing literature, particularly in works involving open data platforms and methodological reporting, whereas reproducibility and collaboration remain underdeveloped. The empirical analyses of RQ5 and RQ7 show that only a small subset of studies provide robust evidence or meaningful conditions for independent verification, underscoring the need for stronger empirical designs and better integration of artifacts, metadata, and execution instructions. The alignment assessment in RQ6 reinforces this imbalance by demonstrating that current contributions only partially reflect the four UNESCO principles, with reproducibility and collaboration showing minimal support. The research gaps synthesized in RQ8 further illustrate the structural, socio-technical, and educational challenges that must be addressed for Open Science to progress in SE.

Building on these insights, the review articulated a structured research agenda and identified cross-cutting actions intended to support cultural, methodological, infrastructural, and policy-driven transformations. These recommendations emphasize the importance of multidimensional evaluation of Open Science practices, the creation of complete and reusable artifacts, the need for longitudinal and multi-organizational empirical studies, and the integration of socio-technical and educational perspectives into Open Science initiatives. They also highlight the role of conferences, journals, institutions, and funding agencies in promoting incentives, harmonizing expectations, and sustaining long-term stewardship of open scientific outputs.

Overall, this work contributes to consolidating Open Science as a research subject within Software Engineering by mapping current practices, identifying weaknesses, and outlining actionable paths for advancement. As Open Science continues to gain relevance in computing disciplines and across the broader scientific ecosystem, the findings presented here provide a foundation for more rigorous, collaborative, and reproducible research practices in Software Engineering. The study encourages the community to move beyond isolated or tool-centric initiatives and toward coordinated, theory-informed, and empirically grounded efforts that can embed openness into the everyday conduct of SE research and practice.

Authors´Contribution

Both authors collaborated in all steps of this study, including analysis, selection of studies, and writing of the manuscript.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Availability of Data and Materials

Artifacts are available at `https://doi.org/10.5281/zenodo.18235983`, licensed under the **Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License**.

Acknowledgments

The authors thank **PROAP/CAPES**, Brazil, for their support (code 001) and the **Araucária Foundation of Paraná** for funding (Grant #PBA2025201000274).

Edson OliveiraJr thanks **CNPq/Brazil** (Grants #311503/2022-5 and #319296/2025-3).

References

Aaltonen, T., Mikkonen, T., Peltola, H., and Salminen, A. (2014). From mashup applications to open data ecosystems. In *Proceedings of The International Symposium on Open Collaboration*, pages 1–8. ACM. DOI: 10.1145/2641580.2641599.

Ampatzoglou, A., Bibi, S., Avgeriou, P., Verbeek, M., and Chatzigeorgiou, A. (2019). Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. *Information and Software Technology*, 106:201–230. DOI: 10.1016/j.infsof.2018.10.006.

Anchundia, C. E. *et al*. (2020). Resources for reproducibility of experiments in empirical software engineering: Topics derived from a secondary study. *IEEE Access*, 8:8992–9004. DOI: 10.1109/ACCESS.2020.2964587.

Azpeitia, I., Iturrioz, J., and Díaz, O. (2020). Volunteering for linked data wrapper maintenance: A platform perspective. *Information Systems*, 89:101468. DOI: 10.1016/j.is.2019.101468.

Badampudi, D., Wohlin, C., and Petersen, K. (2015). Experiences from using snowballing and database searches in systematic literature studies. In *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*, pages 1–10. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/2745802.2745818.

Baltes, S. and Ralph, P. (2022). Sampling in software engineering research: A critical review and guidelines. *Empirical Software Engineering*, 27(4):94. DOI: 10.1007/s10664-021-10072-8.

Basili, V. R., Zelkowitz, M. V., Sjøberg, D. I., Johnson, P., and Cowling, A. J. (2007). Protocols in the use of empirical software engineering artifacts. *Empirical Software Engineering*, 12(1):107–119. DOI: 10.1007/s10664-006-9030-4.

Brito, K. S., da Silva Costa, M. A., Garcia, V. C., and de Lemos Meira, S. R. (2014). Brazilian government open data: implementation, challenges, and potential opportunities. In *Proceedings of the 15th annual international conference on digital government research*, pages 11–16. ACM. DOI: 10.1145/2612733.2612770.

Cito, J. and Gall, H. C. (2016). Using docker containers to improve reproducibility in software engineering research. In *Proceedings of the 38th international conference on software engineering companion*, pages 906–907. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/2889160.2891057.

Cordasco, G., Malandrino, D., Pirozzi, D., Scarano, V., and Spagnuolo, C. (2018). A layered architecture for open data: Design, implementation and experiences. In *Proceedings of the 11th International Conference on Theory and Practice of Electronic Governance*, pages 371–381. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/3209415.3209466.

Cordeiro, A. and OliveiraJr, E. (2025). Foster taxonomy-based open science-related practices in software engineering: Review and observations. In *Brazilian Symposium on Software Engineering*, pages 1–12, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbes.2025.9563.

Correa, A. S., Zander, P.-O., and Da Silva, F. S. C. (2018). Investigating open data portals automatically: a methodology and some illustrations. In *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, pages 1–10. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/3209281.3209292.

Damasceno, C., Melo, I., and Strüber, D. (2021). Towards multi-criteria prioritization of best practices in research artifact sharing. In *Proceedings of the I Workshop on Open Science Practices for Software Engineering (OpenScienSE)*, pages 1–6, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/opensciense.2021.17137.

Dennis, J. B. (2017). Principles to support modular software construction. *Journal of Computer Science and Technology*, 32:3–10. DOI: 10.1007/s11390-017-1702-6.

El Bacha, O., Jmad, O., El Bouzekri El Idrissi, Y., and Hmina, N. (2017). Exploiting open data to improve the business intelligence & business discovery experience. In *Proceedings of the 2nd international Conference on Big Data, Cloud and Applications*, pages 1–6. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/3090354.3090382.

Engström, E., Storey, M.-A., Runeson, P., Höst, M., and Baldassarre, M. T. (2020). How software engineering research aligns with design science: a review. *Empirical Software Engineering*, 25:2630–2660. DOI: 10.1007/s10664-020-09818-7.

Enríquez-Reyes, R., Cadena-Vela, S., Fuster-Guilló, A., Mazón, J.-N., Ibáñez, L. D., and Simperl, E. (2021). Systematic mapping of open data studies: Classification and trends from a technological perspective. *IEEE Access*, 9:12968–12988. DOI: 10.1109/ACCESS.2021.3052025.

Ernst, N. A. and Baldassarre, M. T. (2023). Registered reports in software engineering. *Empirical Software Engineering*, 28:1–11. DOI: 10.1007/s10664-022-10277-5.

Ernst, N. A., Carver, J. C., Mendez, D., and Torchiano, M. (2021). Understanding peer review of software engineering papers. *Empirical Software Engineering*, 26:1–29. DOI: 10.1007/s10664-021-10005-5.

European Organization For Nuclear Research and OpenAIRE (2013). Zenodo. DOI: 10.25495/7GXK-RD71.

Farnham, A., Kurz, C., Öztürk, M. A., Solbiati, M., Myllyntaus, O., Meekes, J., Pham, T. M., Paz, C., Langiewicz, M., Andrews, S., *et al*. (2017). Early career researchers want open science. *Genome biology*, 18:1–4.

DOI: 10.1186/s13059-017-1351-7.

Figshare (2024). Figshare. Available at:`https://figshare.com/`.

Flint, S. (2009). A conceptual model of software engineering research approaches. In *2009 Australian Software Engineering Conference*, pages 229–236. IEEE, IEEE. DOI: 10.1109/ASWEC.2009.42.

Foster (2022). Foster open science taxonomy. Available at:`https://www.fosteropenscience.eu/foster-taxonomy/open-science?page=6`.

Frachtenberg, E. (2022). Research artifacts and citations in computer systems papers. *PeerJ Comput Sci*, 7:e887. DOI: 10.7717/peerj-cs.887.

Frakes, W. B. and Kang, K. (2005). Software reuse research: Status and future. *IEEE transactions on Software Engineering*, 31(7):529–536. DOI: 10.1109/TSE.2005.85.

Ghazi, P. and Glinz, M. (2017). Challenges of working with artifacts in requirements engineering and software engineering. *Requirements Engineering*, 22(3):359–385. DOI: 10.1007/s00766-017-0272-z.

González-Barahona, J. M. and Robles, G. (2012). On the reproducibility of empirical software engineering studies based on data retrieved from development repositories. *Empirical Software Engineering*, 17(1):75–89. DOI: 10.1007/s10664-011-9181-9.

Graziotin, D., Wang, X., and Abrahamsson, P. (2014). A framework for systematic analysis of open access journals and its application in software engineering and information systems. *Scientometrics*, 101:1627–1656. DOI: 10.1007/s11192-014-1278-7.

Grechanik, M., McMillan, C., Dasgupta, T., Poshyvanyk, D., and Gethers, M. (2014). Redacting sensitive information in software artifacts. In *Proceedings of the 22Nd International Conference on Program Comprehension*, pages 314–325. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/2597008.2597138.

Jagtiani, J. L. and Lewis, N. (2016). Enhancing software engineering curricula by incorporating open, data-driven planning methods. In *2016 ASEE Annual Conference & Exposition*, pages 1–11. ASEE PEER. DOI: 10.18260/p.26702.

Juristo, N. and Gómez, O. S. (2010). Replication of software engineering experiments. In *Empirical software engineering and verification*, pages 60–88. Springer. DOI: 10.1007/s10664-013-9290-8.

Khalil, H. and Tricco, A. C. (2022). Differentiating between mapping reviews and scoping reviews in the evidence synthesis ecosystem. *Journal of Clinical Epidemiology*, 149:175–182. DOI: 10.1016/j.jclinepi.2022.05.012.

Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, EBSE Technical Report, Keele University and Durham University. Available at:`https://legacyfileshare.elsevier.com/promis_misc/525444systematicreviewsguide.pdf`.

Kitchenham, B. A., Budgen, D., and Brereton, P. (2015). *Evidence-based software engineering and systematic reviews*, volume 4. CRC Press, CRC Press.

Krishnamurthi, S. (2013). Artifact evaluation for software conferences. *ACM SIGSOFT Software Engineering Notes*, 38(3):7–10. DOI: 10.1145/2464526.2464530.

Krueger, C. W. (1992). Software reuse. *ACM Computing Surveys (CSUR)*, 24(2):131–183. DOI: 10.1145/130844.130856.

Kuhn, A. (2012). Ides need become open data platforms (as need languages and vms). In *2012 Second International Workshop on Developing Tools as Plug-Ins (TOPI)*, pages 31–36. IEEE, IEEE. DOI: 10.1109/TOPI.2012.6229807.

Lopes, A., Conte, T., and Sieckenius de Souza, C. (2020). Exploring the directives of communicability for improving the quality of software artifacts. In *19th Brazilian Symposium on Software Quality*, pages 1–10. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/3439961.3439974.

Lopes Damian, A., Sieckenius de Souza, C., and Conte, T. (2021). Directives of communicability for software artifacts. In *XX Brazilian Symposium on Software Quality*, pages 1–9. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/3439961.3439974.

Lorey, T., Ralph, P., and Felderer, M. (2022). Social science theories in software engineering research. In *Proceedings of the 44th International Conference on Software Engineering*, pages 1994–2005. DOI: 10.1145/3510003.3510076.

Manghi, P., Bardi, A., Atzori, C., Baglioni, M., Manola, N., Schirrwagen, J., Principe, P., Artini, M., Becker, A., De Bonis, M., *et al.* (2019). The openaire research graph data model. *Zenodo*, 1(1):1–23. DOI: 10.5281/zenodo.2643199.

Mauerer, W. and Scherzinger, S. (2022). 1-2-3 reproducibility for quantum software experiments. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 1247–1248. IEEE, IEEE. DOI: 10.1109/SANER53432.2022.00148.

Mendez, D., Graziotin, D., Wagner, S., and Seibold, H. (2020). Open science in software engineering. In *Contemporary empirical methods in software engineering*, pages 477–501. Springer, New York. DOI: `https://doi.org/10.1007/978-3-030-32489-6_17`.

Méndez Fernández, D., Böhm, W., Vogelsang, A., Mund, J., Broy, M., Kuhrmann, M., and Weyer, T. (2019a). Artefacts in software engineering: a fundamental positioning. *Software & Systems Modeling*, 18(5):2777–2786. DOI: 10.1007/s10270-019-00714-3.

Méndez Fernández, D., Monperrus, M., Feldt, R., and Zimmermann, T. (2019b). The open science initiative of the empirical software engineering journal. *Empirical Software Engineering*, 24(3):1057–1060. DOI: 10.1007/s10664-019-09712-x.

Mosconi, G., Li, Q., Randall, D., Karasti, H., Tolmie, P., Barutzky, J., Korn, M., and Pipek, V. (2019). Three gaps in opening science. *Computer Supported Cooperative Work (CSCW)*, 28(3):749–789. DOI: 10.1007/s10606-019-09354-z.

Munn, Z., Peters, M. D., Stern, C., Tufanaru, C., McArthur, A., and Aromataris, E. (2018). Systematic review or scoping review? guidance for authors when choosing between a systematic or scoping review approach. *BMC medical*

research methodology, 18(18):1–7. DOI: 10.1186/s12874-018-0611-x.

Nicolás, J. and Toval, A. (2009). On the generation of requirements specifications from software engineering models: A systematic literature review. *Information and Software Technology*, 51(9):1291–1307. DOI: 10.1016/j.infsof.2009.04.001.

OliveiraJr, E., Madeiral, F., Santos, A. R., von Flach, C., and Soares, S. (2024). A vision on open science for the evolution of software engineering research and practice. In *ACM International Conference on the Foundations of Software Engineering*, page 512–516, New York, NY, USA. ACM. DOI: 10.1145/3663529.3663788.

Olsson, H. H. and Bosch, J. (2014). From opinions to data-driven software r&d: A multi-case study on how to close the'open loop'problem. In *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 9–16. IEEE, IEEE. DOI: 10.1109/SEAA.2014.75.

Open Science, F. (2022). Foster open science. Available at:`https://www.fosteropenscience.eu`.

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and software technology*, 64:1–18. DOI: 10.1016/j.infsof.2015.03.007.

Pfeiffer, R.-H. (2020). What constitutes software? an empirical, descriptive study of artifacts. In *Proceedings of the 17th International Conference on Mining Software Repositories*, pages 481–491. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/3379597.3387442.

Pontika, N., Knoth, P., Cancellieri, M., and Pearce, S. (2015). Fostering open science to research using a taxonomy and an elearning portal. In *Proceedings of the 15th international conference on knowledge technologies and data-driven business*, pages 1–8. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/2809563.2809571.

Ralph, P., Chiasson, M., and Kelley, H. (2016). Social theory for software engineering research. In *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*, pages 1–11. DOI: 10.1145/2915970.2915998.

Ruiz-Rube, I., Dodero, J. M., and Colomo-Palacios, R. (2015). A framework for software process deployment and evaluation. *Information and Software Technology*, 59:205–221. DOI: 10.1016/j.infsof.2014.12.001.

Schlagwein, D., Conboy, K., Feller, J., Leimeister, J. M., and Morgan, L. (2017). "openness" with and without information technology: A framework and a brief history. *Journal of Information Technology*, 32(4):297–305. DOI: 10.1057/s41265-017-0049-3.

Silveira, L. d., Sena, P. M. B., Ribeiro, N. C., Cortés, J. P., Melero, R., Fachin, J., Mora-Campos, A., Santillán-Aldana, J., Piraquive-Piraquive, D. F., Silva, F. C. C. d., *et al*. (2023). Taxonomy of open science: revised and expanded. *Encontros Bibli*, 28:e91712. DOI: 10.5007/1518-2924.2023.e91712/53421.

Söderberg, E., Church, L., and Höst, M. (2021). Open data-driven usability improvements of static code analysis and its challenges. In *Evaluation and Assessment in Software Engineering*, pages 272–277. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/3463274.3463808.

Sommerville, I. (2016). *Software testing*. Pearson Addison-Wesley, Boston, Massachusetts, EUA. Book.

Standard, I. I. (2017). Iso/iec/ieee international standard - systems and software engineering–vocabulary. *ISO/IEC/IEEE 24765:2017(E)*, 1(1):1–541. DOI: 10.1109/IEEESTD.2017.8016712.

Timperley, C. S., Herckis, L., Le Goues, C., and Hilton, M. (2021). Understanding and improving artifact sharing in software engineering research. *Empirical Software Engineering*, 26:1–41. DOI: 10.1007/s10664-021-09973-5.

Traverso-Ribón, I., Ruíz-Rube, I., Dodero, J. M., and Palomo-Duarte, M. (2013). Open data framework for sustainable assessment in software forges. In *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics*, pages 1–8. ACM. DOI: https://dl.acm.org/doi/10.1145/2479787.2479817.

UNESCO, U. (2022). Unesco recommendation on open science. `https://en.unesco.org/science-sustainable-future/open-science/recommendation`.

Verdecchia, R., Engström, E., Lago, P., Runeson, P., and Song, Q. (2023). Threats to validity in software engineering research: A critical reflection. *Information and Software Technology*, 164:107329. DOI: 10.1016/j.infsof.2023.107329.

Vicente-Saez, R. and Martinez-Fuentes, C. (2018). Open science now: A systematic literature review for an integrated definition. *Journal of business research*, 88:428–436. DOI: 10.1016/j.jbusres.2017.12.043.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, pages 1–10. ACM. DOI: https://dl.acm.org/doi/abs/10.1145/2601248.2601268.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media, Germany. Book.