A Sustainable Data Extraction Documentation Process for Literature Studies

Angelika Kaplan
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
angelika.kaplan@kit.edu

Marco Konersmann
Software Engineering, RWTH Aachen University
Aachen, Germany
konersmann@se-rwth.de

Thomas Kühn
Martin-Luther-Universität Halle-Wittenberg
Halle (Saale), Germany
thomas.kuehn@informatik.uni-halle.de

Raffaela Mirandola
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
raffaela.mirandola@kit.edu

Ralf Reussner
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
reussner@kit.edu

Abstract

Background. Open Science advocates for transparency of research outcomes; however, software engineering (SE) research often lacks adequate availability of essential artifacts. This deficiency hinders reproducibility, sustainability, and the broader adoption of Open Science practices. Evidence-Based Software Engineering (EBSE) emphasizes the systematic use of empirical evidence to evaluate SE practices. Systematic literature studies form a cornerstone of EBSE but face challenges in persistency, particularly for data extraction results in a multi-reviewer context, due to limited machine-readable formats and inefficiencies in reviewer collaboration. *Objective.* This paper introduces a practical guideline and documentation process for conducting systematic literature studies with a categorical classification objective (e.g., scoping studies). The approach extends existing methodologies by integrating tool support in terms of version control and machine-readable formats, thereby reinforcing Open Science principles. *Method.* We performed a feature analysis, following the framework of Marshall et al., to evaluate the suitability of selected tools for supporting documentation and coordination in multi-reviewer data extraction scenarios. *Results.* Our proposal for a tool-supported data extraction achieves a high score in the feature analysis, indicating suitability and enhancement of common practices. Our guideline improves the documentation of data extraction results in literature studies and addresses key challenges related to traceability, consistency, and reviewer coordination. *Conclusion.* Research artifacts are critical for the sustainability of SE research and the advancement of Open Science. Our proposed guideline provides a structured, tool-supported approach that enhances transparency, reproducibility, and efficiency in EBSE studies.

CCS Concepts

• **General and reference → Computing standards, RFCs and guidelines**; **Empirical studies**.

WSESE '26, Rio de Janeiro, Brazil

ACM ISBN 979-8-4007-2382-7/26/04
https://doi.org/10.1145/3786149.3788301

Keywords

Open Science, Empirical Software Engineering Research, Sustainable Systematic Literature Study, Data Extraction Process, Collaborative Review, Reproducibility

ACM Reference Format:
Angelika Kaplan, Marco Konersmann, Thomas Kühn, Raffaela Mirandola, and Ralf Reussner. 2026. A Sustainable Data Extraction Documentation Process for Literature Studies. In *3rd International Workshop on Methodological Issues with Empirical Studies in Software Engineering (WSESE '26), April 12–18, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3786149.3788301

1 Introduction

The principles of the Open Science movement (cf. Méndez et al. [12] and Kazman et al. [6]) advocate, among others, for transparency and open accessibility of scientific research outcomes, including results, intermediate steps, and associated artifacts. In software engineering (SE) research, however, the availability of essential research artifacts is often insufficient, limiting maturity and sustainability in scientific research (see dos Santos et al. [3] and Konersmann et al. [10]). Research artifacts, such as papers and supplementary materials, play a crucial role in the sustainable documentation, validation, and dissemination of research knowledge. Thus, research artifacts as digital outputs of research processes and their corresponding documentation offer immense value for both researchers and practitioners.

In this paper, we present a practical guideline and documentation process for systematic literature studies supporting the data extraction aimed at categorical objectives. This includes, in particular, the classification of research papers based on predefined or exploratively developed schemas derived from metadata specifications. Our approach sustainably supports dedicated tasks for the data extraction in a large-scale reviewer context (i.e., reviewer in terms of data extractor and participant in the literature study). It aims to complement existing conceptual guidelines (see Kitchenham and Charters [7] in Section 2.1) by providing instructions for review organizers (leading authors, cf. Section 3.1) and data extractors (cf. Section 3.2) with dedicated tool-support in terms of version control and machine-readable formats (Section 2.4).

This guideline is grounded in our experience and its feasibility has been demonstrated through community acceptance during the review process of our previous work (see Konersmann et al. [10]). Furthermore, we conduct a feature analysis (Section 4.1) according

to Marshall et al. [11] to assess the suitability of the selected tools incorporated in our guideline. This assessment focuses on the documentation support during data extraction within a collaborative, large-scale reviewer setting.

Therefore, we ask the following research question (RQ) along with a validation question (VQ):

RQ: How to support open science principles in literature studies to ensure transparency and replicability in the data extraction phase, especially in a large-scale reviewer context, and how to manage such a process?

VQ: How suitable are the selected tools for our proposed approach according to the feature analysis set of Marshall et al. [11]?

The remainder is structured as follows: In the following, we present background information to our proposed guideline. This includes a complementary methodological guideline to our approach, Open Science principles and the concept of sustainable literature studies, and information about the tools that we use in our guideline (Section 2). Next, we present our approach and guideline for a tool-supported data extraction from an organizer (leading author) and reviewer perspective in Section 3. In Section 4, we present the results of the feature analysis according to Marshall et al. [11] to validate our approach, before we conclude in Section 5.

2 Foundations

In this section, we provide foundations and motivational needs for our work. First, we start with a methodological guideline for systematic literature studies in SE (Section 2.1) and highlight the need for practical, tool-supported guidelines, specifically in the data extraction, to support Open Science principles (Section 2.2) and sustainable literature studies (Section 2.3). Finally, we close this section with an overview of tools (Section 2.4) that we use in our proposed guideline (Section 3)

2.1 Methodological Guidelines for Systematic Literature Studies

The methodological guideline by Kitchenham and Charters [7] is widely used for conducting systematic literature studies in SE. Their framework outlines a structured process comprising three main phases: planning, conducting, and reporting. Within the conduction phase, key activities include study selection, quality assessment, data extraction, and synthesis. The guideline emphasizes methodological rigor and provides conceptual instructions for each phase, including templates and checklists.

However, while the guideline offers a solid conceptual foundation, it remains relatively high-level and does not fully address the practical complexities of executing data extraction in large-scale, multi-reviewer settings. In such contexts, challenges arise related to coordination, consistency, traceability, and documentation. Reviewers often work asynchronously, across institutions, and with varying levels of experience, which can lead to inconsistencies in classification, metadata interpretation, and artifact handling.

To address these gaps, there is a clear need for **practical, tool-supported guidelines** (see Section 3) that complement the conceptual framework of Kitchenham and Charters. Specifically, the data extraction phase requires:

- Structured workflows for assigning and tracking reviewer tasks.
- Version control and traceability mechanisms to manage intermediate results and resolve classification conflicts.
- Machine-readable formats for storing, processing, and synthesizing extracted data.
- Collaborative infrastructure to support communication, consensus building, and documentation.

Such practical support not only enhances reproducibility and transparency but also aligns with the principles of Open Science, FAIR and sustainable research data management. By operationalizing the data extraction process, especially in multi-reviewer environments, we can ensure that systematic literature studies are not only methodologically sound but also scalable, efficient, and adaptable to evolving research practices.

2.2 Open Science in Software Engineering and Provision of Datasets

Empirical software engineering increasingly relies on the availability of structured, reusable datasets to support transparency, reproducibility, and cumulative knowledge building. As highlighted by Kazman et al. [6], the lack of accessible, domain-specific datasets, limits the ability to replicate studies, validate findings, and conduct longitudinal analyses. This challenge is not unique to Software architecture research but reflects a broader issue across the SE community. Daniel Mendez's work [12] on Open Science in Software Engineering emphasizes the need for a cultural and infrastructural shift toward openness, including the public sharing of research artifacts, protocols, and data. To foster Open Science in SE, the community have to prioritize the creation and maintenance of curated datasets, adopt FAIR principles, and encourage the use of platforms that support artifact sharing and traceability. Doing so will enhance the scientific rigor of empirical studies, facilitate cross-study comparisons, and strengthen the connection between academic research and industrial practice, as intended by EBSE research [8].

2.3 Sustainable Literature Studies

dos Santos et al. [3, 4] propose a shift toward sustainable literature reviews as a methodological advancement in SE research. Traditional systematic literature studies are often static, labor-intensive, and difficult to update or replicate. In contrast, sustainable literature reviews aim to ensure long-term usability, transparency, and reproducibility by addressing economic, social, and technical dimensions. The social dimension emphasizes collaboration, stakeholder engagement, and open access to results, promoting inclusivity and transparency. The economic dimension focuses on conserving resources by improving documentation, tool support, and adherence to guidelines to reduce redundant effort. The technical dimension highlights the role of automation and infrastructure in enabling reliable updates and replications. Sustainable literature reviews are conceptualized as evolving, reusable artifacts rather than one-time efforts. This involves the use of machine-readable formats, version control systems (e.g., Git), and automation tools to integrate new evidence continuously. Furthermore, openness and collaboration are encouraged through the public sharing of protocols, datasets, and analysis scripts.

In alignment with these principles, our proposed guideline supports the data extraction phase in large-scale reviewer settings by leveraging machine-readable formats and version control, thereby contributing to the sustainability of literature studies.

2.4 Tool Support and Machine-readable Formats for Literature Studies

In the following section, we present two tools utilized in our approach: the SLR Toolkit (Section 2.4.1) and the distributed version control system Git (Section 2.4.2).

These tools can support the data extraction phase in systematic literature studies involving multiple reviewers. By reducing reviewer errors and enhancing coordination, they contribute to improved effectiveness and efficiency. Moreover, they facilitate the long-term preservation and traceability of extraction results, thereby strengthening the overall reliability and reproducibility of the literature study.

2.4.1 The SLR Toolkit. The SLR Toolkit[1] [5] is an open-source and an Eclipse Rich Client Platform application developed by Sebastian Götz and collaborators to support literature reviews in computer science and software engineering. It streamlines and partially automates the review process by addressing methodological and organizational challenges in the data extraction phase. The toolkit provides an integrated environment for managing workflows, classification schemas, and artifact metadata, offering core functionalities that facilitate efficient and reproducible literature studies. The core features of the SLR toolkit that we utilize for our approach are:

Taxonomy Modeling. A key feature of the SLR Toolkit is its support for designing and applying domain-specific hierarchical taxonomies to classify papers. Reviewers can assign papers to one or more categories using checkboxes, thereby preventing typos. This functionality promotes consistent coding and facilitates collaboration in multi-reviewer settings.

Artifact Management. The toolkit supports the structured storage, annotation, and retrieval of research artifacts, including bibliographic metadata and classification results of reviewers.

The SLR Toolkit is well-suited to our approach, offering model-driven support for defining and reusing classification schemas. It enables structured storage, annotation, and retrieval of bibliographic and classification data (metadata extraction), and facilitates collaborative review with Git integration.

2.4.2 Version Control Systems. Version Control Systems (VCSs) are fundamental tools in software engineering and research, enabling management of changes to digital artifacts such as code, datasets, and documentation). Our guideline incorporates GitLab[2], particularly for supporting the data extraction phase and replication package development in systematic literature studies. At their core, VCSs provide relevant mechanisms in a research context to:

- Track changes: Every modification to a file is recorded with metadata, including the author, timestamp, and a descriptive (commit) message.
- Restore previous versions: Users can revert to earlier states of a file or project.
- Enable parallel development: Multiple contributors can work on different parts of a project simultaneously, with tools to merge changes and resolve conflicts.
- Support branching and tagging: Branches allow for isolated development of features or experiments, while tags mark specific versions, such as releases or publication-ready states.

Git has become the de facto standard for version control due to its flexibility, performance, and integration with platforms like GitHub and GitLab. For our guideline, GitLab is used to support versioning and tracking during the data extraction phase of systematic literature studies. GitLab offers both cloud-hosted and self-managed deployment options, making it suitable for institutional environments with strict data governance requirements. In empirical software engineering and reproducibility studies, both platforms serve as repositories for replication packages and datasets.

3 Our Proposal for a Tool-supported Documentation Process in the Data Extraction

In this section, we present our approach and guideline for a tool-supported documentation process tailored to the data extraction phase of systematic literature studies. The approach is designed to facilitate large-scale review processes, particularly in globally distributed reviewer settings. It addresses our main research question (RQ) by providing structured support for coordination, consistency, transparency, and traceability. Each intermediate result is documented according to a predefined classification and data schema, forming the basis for subsequent data synthesis in the literature review process. In the following, we describe the guideline from both the leading author's perspective (Section 3.1) and the reviewer's perspective (Section 3.2). In addition to the conceptual description, we provide an open-access repository that serves as a template[3]. The repository illustrates the practical implementation of the process, covers additional aspects (such as study selection to support a more generalizable process model), and facilitates reuse by researchers.

3.1 Our Approach from the Perspective of Leading Authors

The following steps and instructions are described for and from the perspective of a leading author (organizer of the literature study) .

Create a Taxonomy File. The SLR Toolkit enables to define the metadata schema for data extraction in a `.taxonomy` file (cf. Section 2.4.1). Reviewers have to adhere to this specification by selecting and labeling included papers accordingly, ensuring structured and consistent classification throughout the review process.

Create a GitLab Project for the Literature Study. Creating a GitLab project is the foundational step for establishing a collaborative environment in our literature study workflow. A GitLab project integrates version control, issue tracking, wikis, and access management. We recommend setting up a GitLab organization to manage multiple studies or assigning individual projects to reviewers.

[1] https://github.com/sebastiangoetz/slr-toolkit [Last accessed on 2025-09-26]
[2] https://docs.gitlab.com/user/ [Last accessed on 2025-09-26]
[3] https://gitlab.com/software-engineering-meta-research/gi-working-group/activities/systematic-literature-study-documentation-process/process-template [Last accessed on 2026-01-26]

The leading author initializes a blank repository, defines key parameters (e.g., project name, description, visibility), and creates a `README` for immediate instruction. This structure supports secure, version-controlled management of all study assets and facilitates coordinated collaboration. The GitLab repository should contain the following structure for the literature study:

- A `.bib` file with entries of the selected papers to investigate that were derived from the study selection and quality assessment process. Each entry have to include a unique BibTeX key following the format `LastnameYYYY(suffix)`, ensuring consistent identification and traceability in the review process.
- A `.taxonomy` file of the SLR Toolkit that facilitates the metadata management and specifies the data extraction format for the reviewers in the literature study.
- A `LICENCE` file under which the repository and data should be published. We recommend publishing the (derived) dataset under an open license `CC BY-SA`[4]).
- A `README` that immediately links to the main page of the respective built-in GitLab wiki of the repository.

After setting these basic steps, the leading author(s) should create a wiki as knowledge base.

Set-up a GitLab Wiki as Knowledge Base and List of Instructions for the Reviewers. The GitLab wiki serves as a centralized knowledge base for reviewers, providing step-by-step instructions and detailed descriptions of the classification schema, including metadata examples. It supports consistent and informed data extraction by offering accessible documentation throughout the review process. For a complete overview of the reviewer workflow, refer to the GitLab wiki[5] of our previous literature study included in our replication package [9].

Collaboration and Group Integration. GitLab supports collaborative literature studies through group-based project sharing and role-based access control. Roles such as *Owner* (for study leaders) and *Developer* (for reviewers) enable structured task management. Additional features, including subgroup organization, merge request approvals, and issue tracking, facilitate hierarchical project coordination and assist consistent workflows.

Repository Branches. After preparing the repository with the relevant content, the leading author should create dedicated Git branches for each reviewer and the final results (`first-reviewer`, `second-reviewer`, ..., `merged`). These branches serve as isolated workspaces, allowing reviewers to work independently and avoid conflicts. The default main branch is protected to ensure the integrity and stability of the repository.

Create Issues / Issue Board and Assign Papers to Reviewers. In addition to serving as a Git server and hosting a built-in wiki, GitLab offers an extensive issue tracking system that supports user assignments and labeling. Within our guideline, we recommend using project-specific labels to categorize issues and monitor the progress of data extraction. This feature enhances coordination and transparency in multi-reviewer literature studies. Therefore, the leading author should create the following labels:

hasFirstReview, *hasSecondReview*, ..., *hasNthReview*, *merged*, *hasNew*, and *helpNeeded*

To manage data extraction in systematic literature studies, each paper is assigned to a GitLab issue named after its BibTeX key (e.g., Kaplan2022). Issues can be created manually or via GitLab's `CSV` import feature, which supports structured metadata such as title, description, assignee, and milestone. This enables efficient bulk creation and assignment of tasks to reviewers. GitLab's issue board provides a visual interface for tracking progress, organizing issues into labeled columns that reflect stages of the data extraction process (e.g., *Open*, *helpNeeded*, *hasFirstReview*, *merged*, *Closed*). This setup supports agile workflows and facilitates structured collaboration among reviewers and study organizers.

Documentation of Data Extraction Results. For the usage of git, we standardized the commit messages. We provided multiple templates to simplify the usage as well as improve traceability. Such templates for the data extractors are provided in Section 3.2.

Support Reviewers. In case of any issues that reviewers might face, they can actively request help from the leading authors. This is facilitated with the *helpNeeded* label. Reviewers have to tag the issue (paper) where the problem occurred or create a new issue with this label and assign it to the GitLab account of the leading authors. It is now the responsibility of the lead authors to provide appropriate support and guidance to the respective reviewer.

Update `.taxonomy` *File if necessary.* During data extraction, reviewers have to report missing metadata or classification elements in the taxonomy by labeling the corresponding GitLab issue with *hasNew*. Upon updating the `.taxonomy` file, the leading author is responsible for notifying all reviewers, preferably via email or GitLab's broadcast messaging feature, to ensure consistent and informed data extraction across the reviewer team.

3.2 Our Approach from the Perspective of Data Extractors

The following steps and instructions are described for and from the perspective of a data extractor / reviewer (participant in the literature study).

Read Instructions in the Wiki and Set-up the Environment. Before beginning data extraction, reviewers have to familiarize themselves with the process by consulting the GitLab wiki prepared by the leading authors. Each reviewer requires a Git account. To initiate the extraction, reviewers download, unpack, and run the SLR Toolkit. They can then import the relevant project repository via Git integration, selecting their assigned branch (e.g., `first-reviewer`, ..., `nth-reviewer`) to work independently.

Select Review Assignment. Each reviewer is (randomly) assigned to specific issues by the leading author, corresponding to the included papers from the study selection. To conduct the data extraction process, they have to select an issue from the list, read the corresponding paper, and start the data extraction / labeling process using the SLR Toolkit.

[4]https://creativecommons.org/licenses/by-sa/4.0/deed.en [Last acc. on 2025-09-27]
[5]https://gitlab.com/SoftwareArchitectureResearch/StateOfPractice/-/wikis/Data-Extraction/Process [Last accessed on 2025-09-27]

Conduct Data Extraction and Labeling. The data extraction process begins with the reviewer selecting an assigned issue in GitLab, which links to the corresponding research paper in PDF format. The reviewer launches the SLR Toolkit, updates the project repository via "Team → Pull", and locates the paper using its BibTeX key. The classification task is performed in the "Taxonomy" view by selecting appropriate leaf nodes, with metadata descriptions available in the project wiki. Upon completion, the reviewer saves the results using "File → Save All", storing the classification in the `.bib` file in a dedicated *classes* entry.

Version Control and Review Submission. Reviewers commit and push their classified labels of a paper and changes using Git version control integrated within the SLR Toolkit to their dedicated branch. Therefore, they publish changes via right click in the n the "Project Explorer" view and select "Team" → "Commit...". Commit messages followed a standardized format:

```
[IssueRef] [Bibkey] merged by [ReviewerName]
```

In this format, *[IssueRef]* is the issue Id (e.g., #10), *[Bibkey]* is the key of the entry in the `.bib` file (e.g., Kaplan2025), and *[ReviewerName]* is the name of the first reviewer (real name or the GitLab account abbreviation).

If the changes are updated in GitLab, reviewers have to label the corresponding issue as *hasFirstReview*, *hasSecondReview*, ..., *hasNthReview* according to their respective assignment in the issue. Once the reviewers have completed their evaluations, the merging process commences.

Consensus Building and Communication. If the reviewers have finished their classification task independently, i.e., an issue is labeled with *hasFirstReview*, *hasSecondReview*, ..., *hasNthReview*, where N is the total number of reviewers per issue, they can discuss each classified paper. For the discussion, they can either (1) hold an (online) meeting for synchronous communication, or (2) use the issue's comments for asynchronous communication.

Review Merging for Final Classification Results. Upon reaching consensus, the first reviewer (leading reviewer of the respective issue) update the classification via the SLR Toolkit in the *Project Explorer*, commit and push the merged review to the `merged` branch for publishing using the format:

```
[IssueRef] [Bibkey] merged by [ReviewerName]
```

Finally, the first reviewers can mark this issue with the label *merged* to complete their responsible tasks.

Visualization of Dataset. Once all papers are classified and reviewer disagreements resolved, the data extraction phase yields a gold standard dataset, which serves as input for the subsequent data synthesis step. The final dataset, exported from the SLR Toolkit as a `.bib` file, contains structured classification results. To enhance interpretability, we developed Visulite [1, 2] as open-access and visualization tool that supports various chart types (e.g., bar and bubble charts) and includes interactive filtering capabilities, facilitating exploration and presentation of the replication package data for various (domain-specific) taxonomies by tool adaption.

4 Validation

In this section, we present our validation results based on a feature analysis according to Marshall et al. [11] by answering our validation question (VQ). First, we present the basic building block and foundation for the feature analysis (Section 4.1), before we discuss the results (Section 4.2).

4.1 Feature Analysis for Tool-supported Literature Studies

To address Open Science and the economic, technical, and social aspects of sustainable literature studies, the integration of automated tool support is essential. Table 1 presents a systematic feature analysis based on the framework by Marshall et al. [11] to evaluate the suitability of selected tools for supporting data extraction and documentation. The feature analysis set consists of *four dimensions*: Economic (F1), Ease of Introduction and Setup (F2), Systematic Review Activity Support (F3), and Process Management (F4). Each dimension is further subdivided into subfeatures, assigned to an ID and weighted by importance levels: Mandatory (M), Highly Desirable (HD), Desirable (D), and Nice to Have (N) with corresponding multipliers. Tools are scored on a scale from 0 (absent) to 1 (fully supported), enabling a structured, quantitative comparison. Weighted scores are aggregated to produce percentage values for each feature set, facilitating comparative evaluation across tools and configurations.

The percentage score for a feature set according to Marshall et al. [11] is determined as:

$$\text{Percentage Score} = \frac{\text{Sum of Weighted Scores}}{\text{Maximum Score}} \times 100 \qquad (1)$$

To enable a normalized assessment of tool performance, the maximum attainable score for each feature set is calculated under the assumption of full feature support. This allows comparison relative to ideal coverage. An overall performance score is then derived by computing a weighted average of the normalized scores across all feature sets, accounting for differences in subfeature counts. The weighting schema (cf. Table 1) prioritizes support for systematic review activities (F3) and process management (F4), reflecting their critical role in tool evaluation. Alternative weighting configurations may be applied to emphasize other aspects, such as usability, depending on the evaluation context.

The overall score [11] for a given tool is calculated using the following formula:

$$\text{Overall Score} = \frac{\sum_{i=1}^{n} (w_i TP_i)}{\sum_{i=1}^{n} (w_i)} \qquad (2)$$

where:

- w_i denotes the weighting assigned to the i^{th} feature set,
- TP_i represents the normalized percentage score for the i^{th} feature set,
- n is the total number of feature sets considered.

To ensure a balanced and context-aware evaluation of tool capabilities, the feature analysis framework by Marshall et al. [11] is applied to assess the tool support used in our proposed guideline (Section 3). While the framework covers the entire literature study process, we focus on the entire Feature Sets F1, F2, and F4,

Table 1: Feature analysis list for systematic literature study tools according to Marshall et al. [11]

Feature Set	ID	Subfeature	Importance	Weighting
F1: Economic	F1-SF01	The tool does not require financial payment to use.	HD (*3)	0.1
	F1-SF02	Maintenance	HD (*3)	
F2: Ease of introduction and setup	F2-SF01	The tool has reasonable system requirements.	M (*4)	0.2
	F2-SF02	Simple installation and setup.	HD (*3)	
	F2-SF03	There is an installation guide.	HD (*3)	
	F2-SF04	There is a tutorial.	HD (*3)	
	F2-SF05	The tool is self-contained.	HD (*3)	
F3: SR activity support	F3-SF06	Data extraction and validation	HD (*3)	0.4
F4: Process management	F4-SF01	Support for multiple users	M (*4)	0.3
	F4-SF02	Document management	M (*4)	
	F4-SF03	Security	D (*2)	
	F4-SF04	Management of roles	HD (*3)	
	F4-SF05	Support for multiple projects	M (*4)	

Table 2: Feature set scores and overall scores

	F1 (score out of 6)		F2 (score out of 16)		F3 (score out of 3)		F4 (score out of 17)		Total (score out of 42)		Overall Score
SLR Toolkit	6	100%	16	100%	2.4	80%	17	100%	41.4	98.57%	92.00%
GitLab	6	100%	16	100%							

and selectively evaluate Feature Set F3 (Systematic Review Activity Support) for the subfeature related to data extraction and validation (F3-SF06). The assessment is based on available documentation, repository data, and the original publication by Götz [5], assuming use of the latest SLR Toolkit release and GitLab.com accounts for all reviewers.

4.2 Results

In the following section, we present our results for each feature set (F1-F4) and the overall results.

4.2.1 Results for Feature Set F1. The economic feature set (F1) evaluates tool-related cost and maintenance aspects. The SLR Toolkit is freely available under the Eclipse Public License and actively maintained, with 25 releases and 23 contributors as of June 2024. GitLab offers multiple deployment options, including a free SaaS version (GitLab.com) suitable for open-source use, and an open-source Community Edition under the MIT License. Both tools follow semantic versioning and maintain regular release cycles, supporting sustainable and cost-effective integration into literature study workflows. In summary, both tools are free to use and well maintained. As a result, they achieve the highest possible score of 6 points each for feature set F1, resulting in a total of 12 points.

4.2.2 Results for Feature Set F2. The Feature Set F2, i.e., the ease of introduction and setup, evaluates usability and setup complexity, including system requirements, installation, documentation, and tool independence. The SLR Toolkit offers a straightforward setup process, supported by tutorials and published documentation. GitLab.com (SaaS) provides a highly accessible, browser-based environment with minimal system requirements and extensive onboarding resources. It is fully self-contained from the user's perspective. Both tools are well-documented, easy to set up, and technically sound, achieving the maximum score for Feature Set F2 by resulting in a total of 32 points.

4.2.3 Results for Feature Set F3. The Feature Set F3 focuses on tool support for activities within the three main phases of a systematic literature study. In our case, we only focus on the data extraction and validation (F3-SF06). The tools should enable structured extraction of qualitative and quantitative data, support classification and mapping, and facilitate collaboration among multiple reviewers. To enhance sustainability, we extend the original framework to include features such as progress tracking, versioning, conflict resolution, and propagation of new information. The SLR Toolkit supports taxonomy-based classification and structured metadata extraction, particularly for qualitative data. It integrates with JBibTeX and Eclipse BIRT for bibliographic and reporting tasks, though quantitative data extraction remains limited. GitLab complements this by offering process tracking, version control, and asynchronous conflict resolution via issue comments. Together, both tools support Open Science principles and collaborative data extraction. However, limitations in quantitative data handling and comment

functionality for the SLR toolkit result in a slightly reduced score (2.4 points) for Feature Set F3.

4.2.4 Results for Feature Set F4. The Feature Set F4 evaluates process management capabilities, including collaboration, document handling, role-based task assignment, and multi-project support. While the SLR Toolkit is primarily designed for individual use, it enables collaborative workflows when combined with Git-based version control. Document organization and metadata management are supported, though security and role management must be handled externally. GitLab.com complements these limitations by offering robust collaboration features, secure document handling, detailed role-based access control, and support for multiple concurrent projects. In combination, both tools fulfill all subfeatures of F4, achieving the highest possible score for this feature set.

4.2.5 Overall Results. Table 2 summarizes the overall results of the feature analysis, showing that both tools, used individually (F1, F2) and in combination (F3, F4), achieved high scores across all evaluated feature dimensions. While the SLR Toolkit demonstrates strong capabilities, there remains potential for improvement, particularly in metadata handling and data extraction specification. Overall, the combined use of the SLR Toolkit and GitLab effectively covers the essential features identified by Marshall et al. [11], supporting the requirements for sustainable literature studies and Open Science.

5 Conclusion and Future Work

In this paper, we presented a tool-supported process for managing classification tasks in systematic literature studies involving multiple data extraction reviewers (Section 3). Therefore, we provided a guideline for literature study organizers and reviewers (participants in the classification tasks). The approach integrates machine-readable formats, version control, issue tracking, and collective documentation via wiki pages to improve the data extraction phase in a multi-reviewer setting. By addressing common challenges such as coordination, traceability, and documentation consistency, the proposed method enhances reproducibility, efficiency and effectiveness in EBSE studies, which tend to be very time-consuming and error-prone. The feasibility of our approach and guidelines has been demonstrated through community acceptance in a peer-review process including artifact evaluation badges awarded in prior work [9]. Furthermore, we conducted a feature analysis (Section 4.2) based on the framework by Marshall et al. [11] to confirm the suitability of the selected tools for supporting collaborative data extraction and transparent documentation to support Open Science and sustainable literature studies. Future work will investigate the integration of artificial intelligence (AI) into the proposed extraction process. Potential applications include using AI as a preliminary classifier to suggest extraction values or as an assistant for consistency and completeness checks. Nevertheless, AI-generated outputs should remain subject to human validation and be explicitly versioned to preserve transparency and reproducibility. Empirical evaluations comparing manual and AI-augmented extraction w.r.t. effort, consistency, and validity are needed to inform evidence-based guidance for the use of AI in literature studies.

Acknowledgments

This work is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under the National Research Data Infrastructure – NFDI 52/1 – project number 501930651, NFDIxCS, supported by funding from the pilot program Core Informatics at KIT (KiKIT) of the Helmholtz Association (HGF) and supported by KASTEL Security Research Labs.

References

[1] Fatma Chebbi, Dominik Fuchß, and Angelika Kaplan. 2023. Visulite Docs. https://github.com/SoftwareArchitectureResearch/visulite

[2] Fatma Chebbi, Dominik Fuchß, and Angelika Kaplan. 2023. Visulite (VISUalization of LITErature). https://softwarearchitectureresearch.github.io/visulite/

[3] Vinicius dos Santos, Anderson Yoshiaki Iwazaki, Kátia Romero Felizardo, Érica Ferreira de Souza, and Elisa Yumi Nakagawa. 2021. Towards Sustainability of Systematic Literature Reviews. In *ESEM '21: ACM / IEEE International Symposium on Empirical Software Engineering and Measurement, Bari, Italy, October 11-15, 2021*, Filippo Lanubile, Marcos Kalinowski, and Maria Teresa Baldassarre (Eds.). ACM, 34:1–34:6. doi:10.1145/3475716.3484192

[4] Vinicius dos Santos, Rick Kazman, and Elisa Yumi Nakagawa. 2025. Leveraging Sustainable Systematic Literature Reviews. *CoRR* abs/2501.01819 (2025). arXiv:2501.01819 doi:10.48550/ARXIV.2501.01819

[5] Sebastian Götz. 2018. Supporting Systematic Literature Reviews in Computer Science: The Systematic Literature Review Toolkit. In *Proceedings of the 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings* (Copenhagen, Denmark) *(MODELS '18)*. Association for Computing Machinery, New York, NY, USA, 22–26. doi:10.1145/3270112.3270117

[6] Rick Kazman, Roberto Tonelli, and Cesare Pautasso. 2023. *An Empirical Basis for Software Architecture Research.* Springer Nature Switzerland, Cham, 87–100. doi:10.1007/978-3-031-36847-9_5

[7] Barabara Kitchenham and Stuart Charters. 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering. https://ebse.webspace.durham.ac.uk/wp-content/uploads/sites/49/2022/08/Systematic-reviews-5-8.pdf [Last accessed on 2025-08-18].

[8] Barbara A. Kitchenham, Tore Dybå, and Magne Jørgensen. 2004. Evidence-Based Software Engineering. In *26th International Conference on Software Engineering (ICSE 2004), 23-28 May 2004, Edinburgh, United Kingdom*, Anthony Finkelstein, Jacky Estublier, and David S. Rosenblum (Eds.). IEEE Computer Society, 273–281. doi:10.1109/ICSE.2004.1317449

[9] Marco Konersmann, Angelika Kaplan, Thomas Kühn, Robert Heinrich, Anne Koziolek, Ralf Reussner, Jan Jürjens, Mahmood Al-Doori, Nicolas Boltz, Marco Ehl, Dominik Fuchß, Katharina Groser, Sebastian Hahner, Jan Keim, Matthias Lohr, Timur Saglam, Sophie Schulz, and Jan-Philipp Töberg. 2022. Replication Package of "Evaluation Methods and Replicability of Software Architecture Research Objects". In *2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C)*. 58–58. doi:10.1109/ICSA-C54293.2022.00021

[10] Marco Konersmann, Angelika Kaplan, Thomas Kühn, Robert Heinrich, Anne Koziolek, Ralf H. Reussner, Jan Jürjens, Mahmood al-Doori, Nicolas Boltz, Marco Ehl, Dominik Fuchß, Katharina Groser, Sebastian Hahner, Jan Keim, Matthias Lohr, Timur Saglam, Sophie Schulz, and Jan-Philipp Töberg. 2022. Evaluation Methods and Replicability of Software Architecture Research Objects. In *19th IEEE International Conference on Software Architecture, ICSA 2022, Honolulu, HI, USA, March 12-15, 2022*. IEEE, 157–168. doi:10.1109/ICSA53651.2022.00023

[11] Christopher Marshall, Pearl Brereton, and Barbara A. Kitchenham. 2014. Tools to support systematic reviews in software engineering: a feature analysis. In *18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14, London, England, United Kingdom, May 13-14, 2014*, Martin J. Shepperd, Tracy Hall, and Ingunn Myrtveit (Eds.). ACM, 13:1–13:10. doi:10.1145/2601248.2601270

[12] Daniel Méndez, Daniel Graziotin, Stefan Wagner, and Heidi Seibold. 2020. Open Science in Software Engineering. In *Contemporary Empirical Methods in Software Engineering*, Michael Felderer and Guilherme Horta Travassos (Eds.). Springer, 477–501. doi:10.1007/978-3-030-32489-6_17