

# An Agentic Approach Towards Replication Package Quality Evaluation

Maximilian Alexander Amougou Mbida   

Technical University of Munich, Germany

Florian Angermeir   

fortiss, Germany

Blekinge Institute of Technology, Sweden

---

## Abstract

Reproducibility in empirical software engineering relies on complete, accessible, and reusable research artifacts, yet artifact evaluation remains largely manual and difficult to scale. This emerging results paper explores an agentic approach for assessing replication package quality by translating open-science guidelines into machine-verifiable criteria. We consolidate 380 requirements from 34 sources into 51 reproducibility criteria, of which 31 are operationalized for automated artifact-based evaluation. Based on these criteria, we implement a multi-agent prototype that inspects replication packages and produces evidence-grounded improvement reports. A preliminary evaluation on five replication packages shows high inter-run consistency of 91.4% and 75.4% correctness, through micro-averaged agreement with a manual baseline. The agent performs best on structural criteria such as code, environment, and artifact availability, but struggles with qualitative or mixed-method studies. A pilot survey with seven software engineering researchers indicates well perceived usefulness and adoption potential, while revealing cognitive load in the human-in-the-loop planning step. Overall, within these small samples, the results indicate that agentic research artifact evaluation has the potential to support authors and reviewers by automating selected routine checks.

**2012 ACM Subject Classification** General and reference → Empirical studies; Software and its engineering → Empirical software validation

**Keywords and phrases** reproducibility, software engineering, research artifacts, agentic systems, multi-agent systems, evaluation framework, research infrastructure

**Digital Object Identifier** 10.4230/LIPIcs.ESEM.2026.58

**Category** Emerging Results, Vision & Reflection Track Paper

**Related Version** *Full Version*: <https://arxiv.org/pdf/2606.02006>

**Supplementary Material** *Software*: <https://doi.org/10.5281/zenodo.20457160>

## 1 Introduction

Reproducibility is a fundamental prerequisite for cumulative scientific progress, enabling independent verification of findings and fostering confidence in published research [17, 6]. In large parts of empirical software engineering, reproducibility depends not only on textual reporting, but also on the availability and quality of accompanying research artifacts such as source code, datasets, execution scripts, configuration files, and experimental environments [24]. To enhance transparent research practices, conferences and publishers have introduced artifact-evaluation tracks, open science guidelines, and reporting standards [18, 2]. Despite these efforts, research artifacts frequently suffer from a plethora of issues, including incomplete source code [16, 12], missing dependencies [23, 13], and fragmented documentation [11]. Consequently, verifying whether these packages are sufficiently complete, accessible, and reusable remains a manual, time-consuming process that scales poorly and is subject to interpretation [25, 24, 9, 21].



© Maximilian Alexander Amougou Mbida and Florian Angermeir;  
licensed under Creative Commons License CC-BY 4.0

20th International Symposium on Empirical Software Engineering and Measurement (ESEM 2026).

Editors: Robert Feldt, Maria Paasivaara, Daniel Mendez, Stefan Wagner, and Marvin Muñoz Barón;

Article No. 58; pp. 58:1–58:13



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

While existing guidelines outline what researchers should report, they provide limited support for automatically assessing the quality of heterogeneous artifact types against predefined criteria representing open science best practices. As a result, artifact evaluation continues to impose substantial effort on reviewers, slows down review processes, and creates barriers for large-scale reproducibility assessments. This challenge is particularly pronounced in contemporary software engineering research, where replication packages lack standardization and often bundle a mix of source code, raw datasets, and configuration files across various directories and formats.

In this paper, we explore whether an agentic approach can support replication package quality evaluation by operationalizing open-science guidelines into machine-actionable verification procedures. We propose an agent-based approach that autonomously retrieves, structures, and evaluates research artifacts against reproducibility criteria, enabling a scalable assessment of research artifacts quality as a proxy to the potential for reproducibility. We target computational, code-centric replication packages containing inspectable artifacts such as source code, datasets, and configuration files. We do not aim to assess the context-dependent rigor of qualitative or theoretical artifacts. Conceptually, our approach can act as a “replication package linter”: it flags evidence-backed quality issues before formal artifact review rather than establishing that a study’s results are reproducible. We hope that our contribution in the long-term can support authors in improving their research artifact quality even before artifact review and to consequently reduce review overhead in artifact review tracks. To investigate the feasibility of this idea, we implemented a prototype based on a multi-agent architecture and conducted a pilot survey. Our preliminary evaluation indicates promising results. The approach achieves an inter-run consistency of 91.4% and a 75.4% micro-averaged agreement with manual baseline. While still preliminary, these results strengthen our confidence in the potential and long-term value of the proposed approach.

The main contributions of this paper are (1) an operationalization of open-science guidelines into a set of machine-verifiable criteria for automated artifact assessment, (2) a novel agentic approach for assessing the reproducibility potential of software engineering research artifacts, and (3) preliminary empirical evidence from a pilot survey that demonstrates the potential and limitations of our approach.

## 2 Related Work

We structure existing related work into challenges with manual and static artifact evaluation, and autonomous reproduction systems.

**Challenges and Automated Artifact Evaluation.** Achieving sustainable reproducibility is hindered by the post-publication fragility of long-term archiving [24] and a pre-publication reliance on manual human labor during peer review. This constraint leads to inconsistent evaluation depth and limits scalability, driving the necessity for automated review assistance. While earlier automation focuses on static analysis of manuscripts, such as Metacheck [8], such solutions can be limited by the lacking of semantic analysis depth of research artefacts. Our approach resolves this through fully automated, end-to-end evaluation of both the artifact content and the structural alignment.

**Agentic Reproducibility Benchmarks and Frameworks.** Recent benchmarks quantify LLM capabilities in scientific reproduction, typically reporting low success rates (20% to 40%) that highlight domain complexity. Evaluation paradigms target either execution and consistency

checks (such as dependency installation in CORE-Bench [21] and paper-to-claim alignment in REPRO-BENCH [14]), or the harder task of generating reproduction code from scratch, as evaluated by PaperBench [22] and LMR-BENCH [26]. To tackle these complexities, several architectures employ specialized reasoning pipelines. For instance, AutoReproduce [27] and ResearchCodeAgent [10] leverage paper lineages and execution feedback to iteratively refine scripts, while SciRep/SciConv [4, 5] automate Docker-based setups. Closest to our work, ArtifactCopilot [25] maps unstructured documentation to dependency graphs for end-to-end ACM badging [9]. However, most of the agentic approaches assume well-structured repositories, complete documentation, or explicit instructions. In contrast, our framework focuses on evaluating the *reproducibility potential* of heterogeneous, inconsistently structured research artifacts where documentation may be partial with the goal of enhancing artifact quality.

### 3 Methodology

To guide the design, implementation, and evaluation of our approach, we organized our methodology around three central research questions:

**RQ1:** How can an agentic approach assess the potential for reproducibility given a research artifact set?

**RQ2:** What are the potential and limitations of an agentic approach to assess the quality of a replication package?

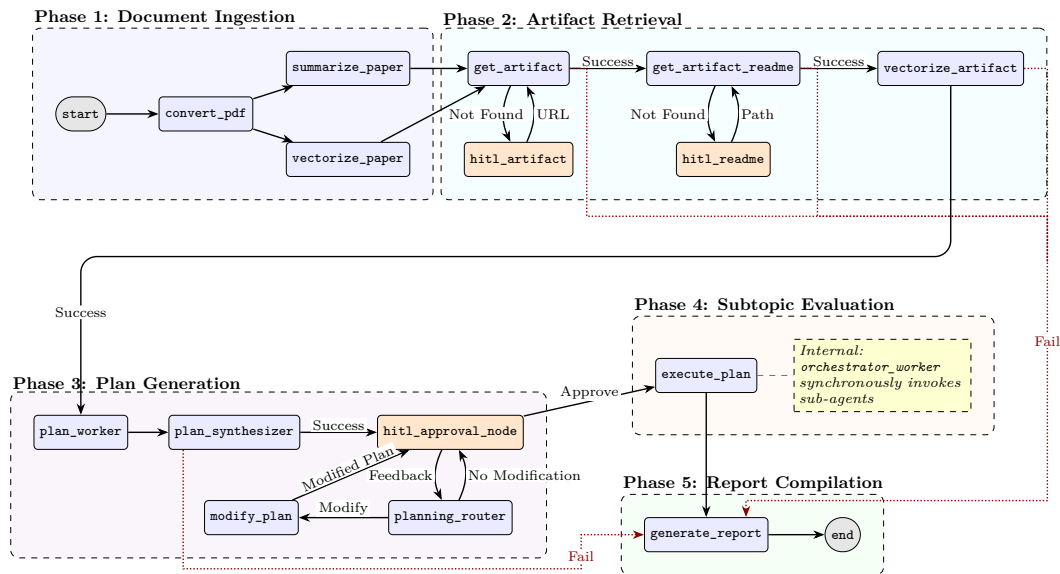
**RQ3:** How useful is the agentic approach for stakeholders assessing replication package quality?

#### Reproducibility Criteria

To establish robust reproducibility criteria, we analyzed official guidelines and reviewer resources from top-tier (CORE A/A\* [3]) computer science conferences featuring artifact evaluation tracks or explicit checklists (e.g., ICSE, OSDI, ICML, RE). This was supplemented by open science policies from major publishers, leading empirical software engineering journals (EMSE, IST, IEEE Access [20]), and community-driven best practices.

**Extraction Process.** We categorized the source documents into four groups: **(1)** conferences with artifact badges, **(2)** checklist-only venues, **(3)** journals and publishers, and **(4)** general guidelines. Verbatim criteria were extracted into a centralized repository, semantically merged within each category, and then consolidated into a unified, hierarchical reproducibility criteria knowledge base. We assigned each criterion a unique identifier to maintain full traceability to its original source, and organized them into hierarchical topics and subtopics. Finally, because raw textual descriptions were insufficient for automated reasoning, we enriched the criteria with structured attributes to enable systematic evaluation by our agent.

For example, the raw requirements “A ReadMe file that includes a table of results accompanied by precise command to run to produce those results” (**fairer\_34**) and “make it easy to reproduce all of the data for your paper, ideally with a single batch script or mouse click” (**aec\_16**) were first normalized into requirements for step-by-step instructions and exact commands. These and related requirements were then merged into the criterion “Provide detailed step-by-step instructions (e.g., in a README) to reproduce all results with exact commands.” We enriched this criterion by specifying the required input as a README, installation or usage documentation, and example commands. The detection strategy checks for sequential instructions and executable command examples and the decision boundary requires a README or installation file containing commands for every experimental step. The database retains links from the merged criterion to all source requirements.



■ **Figure 1** Overview of the five-phase agentic evaluation pipeline. Solid lines indicate standard execution and progression pathways, while dotted lines represent early-exit failure or abort routing back to the report generation phase.

**Enrichment.** Raw textual criteria are often high-level and ambiguous, leading to high evaluation variance that is further exacerbated by LLM non-determinism. To ensure consistent assessment trajectories, we manually enriched each criterion with four structured auxiliary fields: (1) `additional_information` (contextual scope and definitions), (2) `input_needed` (abstract evidence types to seek), (3) `detection_strategy` (initial evaluation heuristics), and (4) `success_failure_conditions` (explicit decision boundaries). We instantiated these specifications in an SQLite knowledge base that preserves relationships and source traceability, and supports relevance-based retrieval for the agent via the full-text search.

## Agentic System Design

The approach was implemented as a stateful, graph-based workflow. As illustrated in Figure 1, the architecture consists of a five-phase pipeline: (1) ingestion and embedding, (2) artifact retrieval and analysis, (3) plan generation, (4) subtopic evaluation, and (5) report compilation. The workflow leverages LangGraph’s Command API for dynamic routing and an `interrupt` mechanism for synchronous Human-in-the-Loop (HITL) interactions, persisting agent state to ensure conversation continuity.

**Phase 1: Ingestion and Embedding.** The input paper is converted from PDF to markdown to extract key metadata (title, abstract, artifact details) and a global summary. Concurrently, the text is chunked and embedded into a vector database for retrieval-augmented generation (RAG) throughout execution.

**Phase 2: Artifact Retrieval and Analysis.** A sub-agent extracts artifact references (e.g., GitHub, Zenodo, OSF) from the paper summary. If automatic retrieval or entrypoint (e.g., README) identification fails, a HITL interrupt requests user guidance, e.g., the correct artifact URL, an upload, or the explicit entrypoint path. If no entry file exists, the agent

logs a failure, and skips to an error report. The retrieved README is summarized to ground planning, a multi-stage filter then prunes the vector index, combining fast rule-based exclusion of irrelevant files (e.g., binaries, dependencies, logs) with an LLM-based classifier for extensionless or ambiguous files. The remaining relevant files are chunked and embedded for hybrid retrieval in the next phase.

**Phase 3: Evaluation Plan Generation.** Worker sub-agents analyze the retrieved context and concurrently generate per-criteria inspection steps for the user-selected subtopics, synthesized into a unified, hierarchical strategy conforming to the knowledge base layout. A HITL loop then lets users iteratively refine this strategy via natural language feedback, suspending execution until explicit user approval.

**Phase 4: Parallel Subtopic Evaluation.** An orchestrator dispatches sub-agents to evaluate each subtopic concurrently within isolated reasoning contexts over a shared, read-only filesystem, gathering evidence via pattern search and hybrid retrieval (lexical and semantic). Evaluation demands evidence-based reasoning with explicit citations (e.g., file paths, code snippets), defaulting to a *fail* status if required evidence is missing. Findings are then synthesized into subtopic- and topic-level summaries and a final structured output.

**Phase 5: Report Compilation and Tracing.** The structured output is transformed into a hierarchical markdown report aggregating evaluation metadata, paper summaries, and detailed criterion assessments linked to their guidelines. For transparency and reproducibility, it logs the complete execution plan and embeds Langfuse trace IDs.

**Frontend.** A web-based prototype exposes four views mirroring the agent’s workflow: configuration, planning (the HITL loop), the execution (real-time reasoning and tool invocations), and the report dashboard.

**Agent Models and Configurations.** The approach uses five agents: artifact retrieval, plan, orchestrator, executor, and reporting. For cost reasons, the retrieval and the executor agent use gpt-5-nano-2025-08-07, the plan agent and orchestrator agents use gpt-5-mini-2025-08-07. The reporting agent uses gpt-5.1-2025-11-13. We set the temperature for all agents to 0. The full agent setup is published in the replication package [1].

## Evaluation

**Reliability and Consistency.** To establish the technical reliability of the automated evaluation, we measured the approach’s *inter-run consistency* and *agreement with a manual baseline* on a small, purposive sample of five replication packages. We selected packages from publications familiar to the authors to support detailed manual assessment. This convenience sample is not representative of replication packages in software engineering, and familiarity may have made the manual assessment easier or more favorable than an assessment of unfamiliar artifacts. Consequently, the resulting scores are exploratory and should not be interpreted as population-level performance estimates.

To evaluate inter-run consistency, we executed the agentic pipeline 10 independent times per package. Stability across identical inputs was evaluated for the execution plan and the evaluation report on a per-element basis for each artifact. We evaluated those elements in two ways. Where possible we used deterministic classification (e.g., Pass/Fail for a subtopic).

Here stability was calculated using a categorical overlap score that aggregates consensus via a bottom-up hierarchical macro-average. Where deterministic classification was impossible (e.g., for the improvement suggestions) we employed an LLM-as-Judge (gpt-5.4-mini-2026-03-17) to evaluate the semantic consistency across randomly paired runs. This judge analyzed the alignment of the agent’s generated outputs on a 1–4 rubric (1 = Not Consistent, 4 = Fully Consistent). We verified the judge’s reliability against a manually created sample, achieving approximately 85% accuracy in score assignment. All scripts, judge prompts, and raw data logs are available in the online material [1].

Second, to quantify the agents agreement with the manual baseline, the authors manually assessed the same five replication packages using the operationalized criteria to establish a structured ground-truth dataset. The agent’s classifications were benchmarked against this manual baseline. To capture granular performance across distinct criteria states (Pass, Fail, Not Applicable), we computed two alignment metrics: a macro-accuracy (to enforce strict schema matching and weights all hierarchical topics equally) and a standard micro-accuracy (to evaluate a flattened intersection of shared criteria, weighting every individual criteria equally). These deterministic metrics were supplemented by the aforementioned LLM-as-Judge, which similarly scored the semantic alignment of the agent’s generated qualitative reports against the human baseline. These measures quantify correspondence with an author-created reference under our operationalized criteria, they do not establish objective correctness, or completeness of the recommendations.

**Perceived Usefulness and Usability.** To assess the practical value of the generated reports, we conducted a survey with active software engineering researchers, ranging from PhD candidates to professors. Participants were recruited via direct email invitations. Prior to interacting with the approach, participants were provided with instructions and context. They were briefed on the prototypes’ objective, provided with credentials for the hosted instance of the prototype, and asked to evaluate one of their own recent research artifacts. Crucially, to ensure data privacy, participants were instructed to avoid uploading confidential or unpublished materials due to the prototypes reliance on commercial LLMs.

Following this setup, participants used the prototype and evaluated their chosen artifact by interacting with the agent’s evaluation plan and reviewing the final report. Participants then completed a structured questionnaire, which was informed by the Technology Acceptance Model (TAM) [7], comprising five-point Likert-scale items and open-ended questions. This allowed us to evaluate the prototypes perceived usefulness and quality (assessing the accuracy, actionability, and detail of the feedback) alongside its usability and adoption potential (ease of use, interaction design, and the likelihood of integration into regular workflows). The survey instrument and anonymized responses are published in the online material [1].

## 4 Emerging Results

This section presents our findings structured around our three research questions: the formulation of the reproducibility criteria (RQ1), an analysis of the agent’s strengths and limitations (RQ2), and an empirical assessment of the approach’s perceived usefulness to stakeholders (RQ3).

### RQ1: Reproducibility Criteria

We distill an initial pool of 380 raw requirements from 34 sources into a unified set of 51 reproducibility criteria. Because some requirements demand paper-text analysis rather than artifact inspection, we distinguish between the full set and a subset of 31 operationalized

criteria that are fully amenable to automated, artifact-based evaluation. The extracted criteria are organized into a hierarchical structure comprising five high-level topics and fourteen subtopics. In the list below, the numbers in parentheses indicate the total count of individual criteria assigned to each respective topic and subtopic:

**Documentation (15):** Covers the core information required to comprehend and reproduce the research. Subtopics: *Theoretical & Mathematical* (7), *Data* (5), and *Experimental & Methodological* (3).

**Data and Artifact Availability (16):** Focuses on ensuring artifacts remain accessible, reusable, and secure over time. Subtopics: *Archiving & Accessibility* (6), *Data Integrity & Privacy* (4), *Licensing & Citation* (2), and *Reproducibility Support* (4).

**Code and Environment (7):** Addresses the executability of the software artifact. Subtopics: *Automatization* (1), *Code Availability & Management* (3), *Integration & Documentation* (1), and *Reproducibility & Infrastructure* (2).

**Experimental Rigor (7):** Captures transparency and reliability of empirical setups. Subtopics: *Experimental Reliability & Statistical Reporting* (2), *Hyperparameter Transparency* (3), and *Model & Randomness Control* (2).

**Ethical and Governance (6):** Encompasses ethical design and reporting considerations. Subtopics: *Human-Subject Research* (2) and *Responsible Use of Models & Code* (4).

## RQ2: Strengths and Limitations

To answer RQ2, we analyze the potential and limitations of using the agentic approach to assess the quality of replication packages.

**Strengths.** The approach shows strong potential in automating the foundational, labor-intensive steps, particularly for computational and code-centric research. Its retrieval pipeline robustly ingests diverse sources, autonomously cloning GitHub repositories, fetching Zenodo archives, and parsing direct URLs or manual uploads.

**Inter-Run Consistency (Stability).** The agent demonstrated high determinism across repeated executions. Using a hierarchical macro-averaging metric that follows our predefined knowledge base schema (i.e., calculating categorical overlap exactly across the established topics, subtopics, and criteria), the approach achieves an overall consistency score of 91.4% across all runs. Stability was particularly strong in structured domains, such as *Data and Artifact Availability* (94.8%) and *Ethical and Governance* (94.0%).

When evaluating via the LLM-as-Judge semantic rubric (scored 1–4), the agent’s high-level qualitative outputs proved similarly stable. Applying this validated judge, the agent’s natural language reasoning (3.36/4.0) and replication improvement suggestions (3.23/4.0) consistently achieve a “Mostly Consistent” rating across runs. Interestingly, the lowest stability is observed in exact evidence path extraction (2.64/4.0), indicating that while the agent’s final reasoning and evaluations are deterministic, the localized text snippets it cites to justify those decisions can vary stochastically between runs.

**Limitations and Failure Cases.** Despite its strengths, testing revealed several limitations and failure cases driven by the tension between rigid automated evaluation and heterogeneous scientific reporting:

**Rigidity and Prompt Engineering Overhead:** To preserve internal consistency, the success and failure boundaries for criteria must be defined very strictly, which reduces flexibility. For example, evaluating whether data was appropriately anonymized initially resulted

in inconsistent penalties for purely technical datasets. Preventing these false negatives required engineering the criteria’s applicability conditions to explicitly distinguish between human-subject data and inherently synthetic data (e.g., regex syntax or physics simulations).

**Contextual Blind Spots (The “If Applicable” Problem):** The approach struggles with conditional criteria. For instance, criteria requiring authors to provide pre-trained model weights or parameter counts caused false failures when the agent evaluated replication studies. Lacking the semantic awareness that the authors were re-running previous code rather than training a novel model, the agent incorrectly demanded non-existent artifacts.

**Extraction and Attribution Ambiguity:** Artifact links frequently hide in footnotes, unstructured data availability sections, or inline citations, and are inconsistently labeled (e.g., “online material” versus “replication package”). This occasionally causes the agent to misclassify a supplementary dataset as a primary code repository.

**Preprocessing and Pipeline Fragility:** Edge cases in formatting can quickly exceed the context window size and processing constraints. During evaluation, oversized or malformed inputs (e.g., data-heavy PDFs, image-laden Jupyter Notebooks, and deeply nested archives) triggered an explosion of text chunks, system timeouts, or an empty workspace with subsequent execution faults. As a preliminary mitigation, we instruct users to provide flattened, non-nested archives prior to uploading their artifacts.

### RQ3: Usefulness for Stakeholders

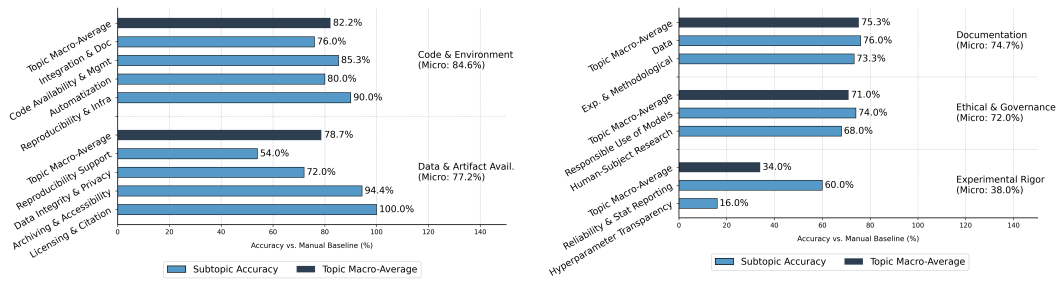
Since a report is only useful if its judgments are correct, we first benchmark the agent against a manual baseline, then survey researchers on perceived usefulness.

**Agreement with Manual Baseline.** When comparing against human-authored baselines, the agent achieved a micro-accuracy of 75.4% and a macro-average of 68.2%. Breaking down the classification metrics reveals performance differences. The agent is reliable at identifying compliant practices, achieving 86.4% precision and 81.5% recall ( $F1 = 0.83$ ) when classifying criteria as “Passed” (True) in the final report. Conversely, when flagging non-compliant (“False”) criteria, it exhibited high recall (85.8%) but lower precision (56.7%), suggesting a slight bias toward strictness, which represents our prompting strategy, instructing the agent to default to false if the answer is not clear.

Crucially, the intersection analysis highlights a massive performance variance depending on the evaluation topic. The agent performs well on structural verifications like *Code and Environment* (84.6%) and *Data and Artifact Availability* (77.2%), but performance plummeted on *Experimental Rigor* (38.0%). See Figure 2 for the full breakdown. Our review indicates that this discrepancy is bidirectional, a point we examine in Section 5.

Finally, qualitative semantic alignment with the human reports reflects this broader performance variance. We used an LLM-as-Judge to score the agent’s semantic alignment against the human-authored ground truth (1–4). The agent achieved a mean alignment score of (2.99/4.00) for improvement suggestions, significantly higher than the (1.92/4.00) observed for evidence paths. We examine this divergence in Section 5.

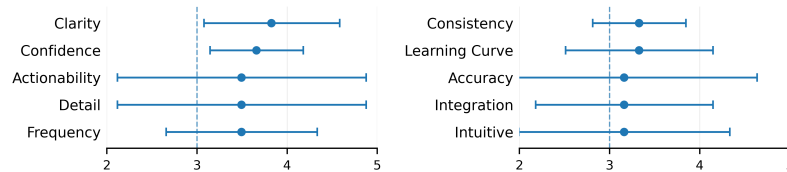
**Pilot Survey.** Across the seven survey participants (3 quantitative, 3 qualitative, and 1 mixed-method study authors), results indicate high usability and strong adoption potential. Within this small pilot sample, participants rated the tool’s clarity and integration positively, with subtopic understanding (Mean, standard deviation  $M = 3.83, SD = 0.75$  Clarity) scoring highest and frequency-of-use intention ( $M = 3.50, SD = 0.84$  Frequency) also high.



■ **Figure 2** Accuracy benchmarked against the manual baseline, detailing both granular subtopic accuracy (blue) and the overall topic macro-average (dark slate).

Perceived usefulness was favorable, as participants felt confident using the tool to improve artifacts ( $M = 3.67, SD = 0.52$  Confidence). However, qualitative consistency and report accuracy showed higher variance ( $M = 3.33, SD = 0.52$  Consistency;  $M = 3.17, SD = 1.47$  Accuracy; respectively), reflecting mixed performance across diverse methodology types. With only seven participants, these means and standard deviations can not be considered stable descriptive summaries and do not support statistical or population-level inference. The study-type counts likewise do not support subgroup comparisons. The qualitative responses provide initial explanations for the high variance. Participants reported that the tool worked better for code-centric, quantitative artifacts than for complex qualitative or mixed-method material. Figure 3 lists the overall ratings and standard-deviations.

**Cost.** Based on the Langfuse traces collected during development and evaluation, the per-paper cost of running the agent ranged from \$0.11 to \$2.23, with an average cost of \$0.71 per report.



■ **Figure 3** Descriptive participant ratings of the prototype ( $n = 7$ ). The five-point Likert scale ranges from 1 (strongly disagree) to 5 (strongly agree), with 3 as the neutral midpoint (vertical line). Points show item means and horizontal bars show  $\pm 1$  standard deviation. Because this is a small pilot sample, the values summarize these participants only and should not be interpreted as population estimates.

## 5 Discussion

The findings from this study demonstrate that agentic approaches can successfully operationalize broad open-science rules into automated checks, particularly for computational and code-centric research. The approach's high inter-run consistency indicates that the inherent non-determinism of LLMs can be effectively constrained through explicit graph-based state management, structured tool use, and operationalized criteria. The proposed approach can handle high heterogeneity of replication packages reasonably well. Furthermore, the architecture introduces robust artifact ingestion that gracefully handles the structural heterogeneity

and format diversity of replication packages. Unlike end-to-end models that generate opaque judgments, the agent strictly grounds its evaluations in extracted evidence. By mandating explicit file-path citations, the approach provides verifiable proof, allowing reviewers to independently audit its reasoning. This is supported by the high semantic consistency scores observed via the LLM-as-Judge evaluation for the agent’s reasoning and suggestions.

The agent’s bias toward strictness, reflected in its lower precision on non-compliant criteria, stems from a reliance on narrowly defined criteria that cause it to penalize artifacts organized in non-standard ways. For instance, when an author provides a unified `reproduce_everything.sh` script rather than the one-to-one figure mapping the agent expects, the approach might flag it as a failure. The same rigidity drives the low agreement on *Experimental Rigor*, where the agent’s sub-condition checks diverge from the heuristics human reviewers apply. The challenge for the future is therefore to move beyond rigid, checklist-based verification and recognize when an alternative implementation is functionally equivalent to the expected standard. Likewise the low semantic agreement on evidence extraction reflects differing retrieval strategies. The agent’s, more exhaustive, retrieval surfaces secondary, corroborating evidence that human reviewers bypass once a sufficient proof is found, paradoxically penalizing its alignment score.

Furthermore, the approach’s limited applicability to qualitative research remains a limitation. Qualitative research relies on context and researcher judgment, expressing rigor through traceability and logical consistency [15], rather than strict re-execution. Future work could explore the integration of structured qualitative reporting frameworks into machine-operationalizable representations. Fortunately, the approach’s modular knowledge base supports this extensibility. By separating the core execution logic from the underlying relational database, the approach can dynamically adapt to venue-specific checklists, publisher mandates, or emerging domain standards (e.g., guidelines for LLM studies [2]) without requiring architecture modifications.

Rather than offering a simple pass-or-fail judgment, the approach diagnoses specific shortcomings and generates a to-do list for artifact improvement. However, user feedback regarding usability underscores a tension between this detailed transparency and cognitive load. As one participant noted regarding the Human-in-the-Loop (HITL) step: “*The planning step expecting human-in-the-loop guidance is overwhelming and it is very unclear to a new user, in how much detail they would need to understand and adjust the plan.*” Future iterations of the frontend would benefit from adaptive interfaces that support progressive disclosure, allowing users to verify high-level decisions while exploring detailed reasoning traces only when explicitly needed. Ultimately, integrating such agentic evaluation approaches has the potential to reshape empirical software engineering research by shifting artifact evaluation from a retrospective, manual process to a scalable, proactive workflow. While confidentiality is crucial when reviewing unpublished work, our prototype relied on commercial models, which is appropriate at this early stage but of course unsuitable in the long run.

## 6 Threats to Validity

We discuss the threats to validity of this study as outlined by Runeson and Höst [19].

**Internal Validity.** Our prompting strategies may favor specific document structures or reporting styles, and model-specific traits, such as phrasing sensitivity, pretraining biases, and stochastic variations that we cannot mitigate within an emerging results report. The use of proprietary LLMs also risks implicit data leakage should the evaluated paper or public

repository appear in their training datasets. We deem this risk as negligible. Because the selected packages came from publications familiar to the authors, this familiarity may have simplified evidence discovery or influenced borderline judgments, thereby distorting measured agreement. We did not measure inter-rater reliability for the manual reference, so its own reliability remains unknown. The reference should therefore be understood as a fallible author-created comparison rather than ground truth.

**External Validity.** Our approach and underlying criteria are systematically biased toward computational, code-centric research, limiting generalizability to qualitative or theoretical software engineering paradigms. In our study, this expected bias stemming from the alignment of the open-science policies we drew on, caused the agent to inappropriately recommend restructuring a qualitative artifact into a code repository. Results also depend on the specific LLM versions used, and neither generalization across LLM versions nor comparison to other models was in scope of this work. The five purposively selected packages are too few to represent the diversity of replication packages in software engineering, and the seven recruited researchers are too few to represent the broader researcher population. Moreover, participants evaluated their own artifacts, so familiarity with and ownership of those artifacts may have influenced perceived usefulness, accuracy, and adoption intentions. We deem those risks acceptable for an emerging results report.

**Construct Validity.** Operationalizing reproducibility through a set of quality criteria may not fully capture its multifaceted nature in practice, a risk we accept as these criteria reflect the current state of practice. Employing an LLM-as-Judge adds a layer of approximation, relying on semantic, prompt-based comparison rather than deterministic verification, which we quantified through manual verification in Section 4.

**Reliability.** The non-deterministic nature of LLMs poses challenges for evaluation consistency. Although our prototype reached a high inter-run consistency, minor stochastic discrepancies persist across repeated executions. We mitigate this by triangulating automated judgements through multi-run aggregation and benchmarking against manual baseline assessments.

## 7 Conclusion

This study demonstrates that agentic workflows have the potential to operationalize high-level open-science policies into machine-verifiable tasks. Our prototype achieved high inter-run consistency and substantial agreement with manual baseline assessments. While the agent excels at offloading labor-intensive tasks (e.g., discovery and structural analysis) for quantitative, code-centric repositories, it exhibits a structural bias against qualitative research and unconventional artifact layouts. While its evidence-based traceability builds user trust, the interactive HITL phase can introduce a noticeable cognitive load, highlighting the need for a more streamlined user experience.

Our three contributions are a relational knowledge base of operationalized open-science criteria, a multi-agent prototype that autonomously retrieves, structures, and evaluates research artifacts, and a pilot survey on the usefulness of the approach. Together they take a step toward shifting artifact evaluation from a manual, resource consuming process, into a scalable, proactive workflow. Ultimately, we hope that in the future this approach can serve authors as a proactive “replication package linter” that flags quality issues prior to peer review, and supports artifact evaluation committees as a force multiplier that automates routine structural checks so reviewers can focus on deep semantic validation.

Future work will harden preprocessing against complex formats such as deeply nested archives and Jupyter Notebooks, integrate study-type detection for adaptive, context-aware criteria selection, apply ensemble methods such as majority voting to further stabilize consistency, and refine the frontend while adding support for open LLMs to broaden adoption. Finally, future work will have to evaluate the impact on resources in artefact evaluation tracks envisioned with our solution.

## Data Availability Statement

The replication package supporting this study, including the scripts, raw repository data, agent configurations, survey instrument and data, is openly available on Zenodo at in our online material [1] under a CC-BY 4.0 license. The replication package contains all materials necessary to reproduce our findings.

## AI Usage

We employed LLMs for textual editing, as well as diagram improvements.

---

## References

- 1 Maximilian Alexander Amougou Mbida and Florian Angermeir. An agentic approach towards replication package quality evaluation. doi:10.5281/zenodo.20457160.
- 2 Sebastian Baltes, Florian Angermeir, Chetan Arora, Marvin Muñoz Barón, Chunyang Chen, Lukas Böhme, Fabio Calefato, Neil Ernst, Davide Falessi, Brian Fitzgerald, Davide Fucci, Marcos Kalinowski, Stefano Lambiase, Daniel Russo, Mircea Lungu, Lutz Prechelt, Paul Ralph, Rijnard van Tonder, Christoph Treude, and Stefan Wagner. Guidelines for empirical studies in software engineering involving large language models, 2025. arXiv:2508.15503.
- 3 CORE. Core conference rankings portal. URL: <https://portal.core.edu.au/conf-ranks/>.
- 4 Lázaro Costa, Susana Barbosa, and Jácome Cunha. A framework for supporting the reproducibility of computational experiments in multiple scientific domains. *arXiv preprint*, 2025. doi:10.48550/arXiv.2503.07080.
- 5 Lázaro Costa, Susana Barbosa, and Jácome Cunha. Let’s talk about it: Making scientific computational reproducibility easy, 2025. doi:10.48550/arXiv.2504.10134.
- 6 Fabio QB Da Silva, Marcos Suassuna, A César C França, Alicia M Grubb, Tatiana B Gouveia, Cleviton VF Monteiro, and Igor Ebrahim dos Santos. Replication of empirical studies in software engineering research: a systematic mapping study. *Empirical Software Engineering*, 19(3):501–557, 2014. doi:10.1007/S10664-012-9227-7.
- 7 Fred D. Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3):319–340, 1989. URL: <http://www.jstor.org/stable/249008>.
- 8 Lisa DeBruine and Daniel Lakens. *metacheck: Check Research Outputs for Best Practices*, 2025. Accessed: 2025-12-15. URL: <https://github.com/scienceverse/metacheck>.
- 9 Association for Computing Machinery. Artifact review and badging: Current, 2020. URL: <https://www.acm.org/publications/policies/artifact-review-and-badging-current>.
- 10 Shubham Gandhi, Dhruv Shah, Manasi Patwardhan, Lovekesh Vig, and Gautam Shroff. Researchcodeagent: An llm multi-agent system for automated codification of research methodologies, 2025. doi:10.48550/arXiv.2504.20117.
- 11 Jesús M González-Barahona and Gregorio Robles. On the reproducibility of empirical software engineering studies based on data retrieved from development repositories. *Empirical Software Engineering*, 17(1):75–89, 2012. doi:10.1007/S10664-011-9181-9.
- 12 Jesus M. Gonzalez-Barahona and Gregorio Robles. Revisiting the reproducibility of empirical software engineering studies based on data retrieved from development repositories. *Information and Software Technology*, 164:107318, 2023. doi:10.1016/j.infsof.2023.107318.

- 13 Zara Hassan, Christoph Treude, Michael Norrish, Graham Williams, and Alex Potanin. Characterising reproducibility debt in scientific software: A systematic literature review. *Journal of Systems and Software*, 222:112327, 2025. doi:10.1016/j.jss.2024.112327.
- 14 Chuxuan Hu, Liyun Zhang, Yeji Lim, Aum Wadhwani, Austin Peters, and Daniel Kang. Repro-bench: Can agentic ai systems assess the reproducibility of social science research?, 2025. doi:10.48550/arXiv.2507.18901.
- 15 Yvonna S Lincoln. *Naturalistic inquiry*, volume 75. sage, 1985.
- 16 Chao Liu, Cuiyun Gao, Xin Xia, David Lo, John Grundy, and Xiaohu Yang. On the reproducibility and replicability of deep learning in software engineering. *ACM Trans. Softw. Eng. Methodol.*, 31(1), October 2021. doi:10.1145/3477535.
- 17 National Academies of Sciences, Medicine, Policy, Global Affairs, Board on Research Data, Division on Engineering, Physical Sciences, Committee on Applied, Theoretical Statistics, Board on Mathematical Sciences, et al. *Reproducibility and replicability in science*. National Academies Press, 2019.
- 18 Paul Ralph, Nauman bin Ali, Sebastian Baltes, Domenico Bianculli, Jessica Diaz, Yvonne Dittrich, Neil Ernst, Michael Felderer, Robert Feldt, Antonio Filieri, Breno Bernard Nicolau de França, Carlo Alberto Furia, Greg Gay, Nicolas Gold, Daniel Graziotin, Pinjia He, Rashina Hoda, Natalia Juristo, Barbara Kitchenham, Valentina Lenarduzzi, Jorge Martínez, Jorge Melegati, Daniel Mendez, Tim Menzies, Jefferson Moller, Dietmar Pfahl, Romain Robbes, Daniel Russo, Nyyti Saarimäki, Federica Sarro, Davide Taibi, Janet Siegmund, Diomidis Spinellis, Mirosław Staron, Klaas Stol, Margaret-Anne Storey, Davide Taibi, Damian Tamburri, Marco Torchiano, Christoph Treude, Burak Turhan, Xiaofeng Wang, and Sira Vegas. Empirical standards for software engineering research, 2020. arXiv:2010.03525.
- 19 Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009. doi:10.1007/s10664-008-9102-8.
- 20 SCImago Lab. Scimago journal & country rank (sjr). URL: [www.scimagojr.com/journalrank.php](http://www.scimagojr.com/journalrank.php).
- 21 Zachary S. Siegel, Sayash Kapoor, Nitya Nagdir, Benedikt Stroebl, and Arvind Narayanan. Core-bench: Fostering the credibility of published research through a computational reproducibility agent benchmark, 2024. doi:10.48550/arXiv.2409.11363.
- 22 Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. Paperbench: Evaluating ai’s ability to replicate ai research, 2025. doi:10.48550/arXiv.2504.01848.
- 23 Ana Trisovic, Matthew K Lau, Thomas Pasquier, and Mercè Crosas. A large-scale study on research code quality and execution. *Scientific Data*, 9(1):60, 2022.
- 24 Stefan Winter, Christopher S. Timperley, Ben Hermann, Jürgen Cito, Jonathan Bell, Michael Hilton, and Dirk Beyer. A retrospective study of one decade of artifact evaluations. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2022, pages 145–156, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3540250.3549172.
- 25 Zhaonan Wu, Yanjie Zhao, Zhenpeng Chen, Zheng Wang, and Haoyu Wang. Agent-based software artifact evaluation, 2026. doi:10.48550/arXiv.2602.02235.
- 26 Shuo Yan, Ruochen Li, Ziming Luo, Zimu Wang, Daoyang Li, Liqiang Jing, Kaiyu He, Peilin Wu, George Michalopoulos, Yue Zhang, Ziyang Zhang, Mian Zhang, Zhiyu Chen, and Xinya Du. Lmr-bench: Evaluating llm agent’s ability on reproducing language modeling research, 2025. doi:10.48550/arXiv.2506.17335.
- 27 Xuanle Zhao, Zilin Sang, Yuxuan Li, Qi Shi, Weilun Zhao, Shuo Wang, Duzhen Zhang, Xu Han, Zhiyuan Liu, and Maosong Sun. Autoreproduce: Automatic ai experiment reproduction with paper lineage, 2025. doi:10.48550/arXiv.2505.20662.