Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

CHOOSING BETWEEN OPEN SOURCE AND PROPRIETARY SOFTWARE: TOTAL COST, SECURITY, GOVERNANCE, AND DIGITAL SOVEREIGNTY

Daniel Henrique Loesch Luis Alves
Faculdade Presidente Antônio Carlos, Brazil
Corresponding author: loeschdaniel07@gmail.com
Orcid ID: https://orcid.org/0009-0001-3780-4024

Abstract
Software selection is frequently reduced to a contest between free access and commercial quality. This integrative narrative review reframes the choice as a lifecycle dependency and control problem. It synthesizes analytical, theoretical, legal, standards, and policy literature on functional fit, total cost of ownership, software supply-chain security, interoperability, governance, sustainability, digital sovereignty, artificial intelligence, and exit capacity. The article formalizes the Lifecycle Dependency and Control Framework for Software Selection (LDCF), which links four theoretical perspectives to six decision domains and distinguishes evidence supplied by empirical research, theory, institutional guidance, and author-generated synthesis. The central proposition is that open source creates option value only when an organization can exercise the rights it receives, whereas proprietary software creates service value only when contractual and technical commitments are credible. Neither licensing model is inherently cheaper, safer, more sustainable, or more sovereign. A defensible decision therefore compares complete deployments over a defined horizon, separates product security from ecosystem and operational security, tests portability, and assigns governance responsibilities. The framework structures context-specific judgment; it is not a universal numerical ranking. Because selection, appraisal, and thematic coding were performed by one author, the synthesis is explicitly interpretive and does not claim inter-rater reliability or exhaustive systematic coverage.
Keywords: open source software; proprietary software; total cost of ownership; software supply chain; vendor lock-in; digital sovereignty; software governance

1. Introduction

Open source software is embedded throughout contemporary digital infrastructure, while proprietary operating systems, applications, platforms, and cloud services

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

remain economically powerful. The resulting landscape is not divided into two self-contained camps. Commercial products routinely incorporate open source components, open source projects depend on corporate labor and infrastructure, and organizations assemble portfolios that cross licensing and delivery models. A comparison at the level of labels therefore hides the dependencies that actually determine cost, security, and control (Fitzgerald, 2006; West, 2003).

Licensing defines permissions rather than performance. An open source license generally permits source inspection, modification, and redistribution under specified conditions, whereas proprietary licensing reserves more control to the copyright holder. These legal differences affect organizational options, but neither model establishes that a product is secure, usable, supported, or appropriate. Such outcomes depend on engineering quality, governance, funding, documentation, integration, and operational practice (Open Source Initiative [OSI], n.d.; Wheeler, 2015).

The debate has acquired strategic importance because governments and firms increasingly treat software dependencies as questions of resilience and sovereignty. The European Commission has described open source as relevant to technological independence, competitiveness, and innovation, while the U.S. Cybersecurity and Infrastructure Security Agency has published a roadmap for strengthening the security of the open source ecosystem (CISA, 2023; European Commission, 2021). These initiatives recognize both the value of shared software commons and the risks created when critical components lack sustained stewardship.

This article asks: under which technical, economic, and institutional conditions does open source or proprietary software provide the stronger lifecycle outcome? It contributes the Lifecycle Dependency and Control Framework for Software Selection (LDCF). The framework operationalizes the article's central thesis: open source rights generate option value only where the adopting organization has the capability to inspect, adapt, maintain, and replace the software; proprietary service commitments generate service value only where technical evidence, enforceable contracts, and credible exit arrangements make those commitments trustworthy. The objectives are to integrate theory with decision criteria, compare lifecycle cost and layered security, examine sovereignty and AI openness, and define boundary conditions for organizational adoption.

2. Literature Review

2.1 From Free Software to Hybrid Production

The modern free software movement framed access to source code as a matter of user freedom, while the later open source movement emphasized a development and licensing model compatible with commercial adoption. This distinction remains important because ethical commitments, engineering practices, and business

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

strategies overlap but are not identical. Organizations may publish code for community benefit, standardization, recruitment, ecosystem growth, or competitive positioning (Lerner & Tirole, 2002; Raymond, 1999).

Open source production challenges the assumption that software must be created inside one firm and sold as a closed artifact. Distributed contributors can identify defects, create extensions, localize interfaces, and reuse components. Yet participation is highly unequal: a small number of maintainers often carry disproportionate responsibility, and corporate contributors may influence project priorities. Openness expands the potential contributor set but does not ensure broad participation or democratic governance (Crowston et al., 2012; Eghbal, 2016).

Proprietary production centralizes product authority, roadmap control, and release decisions. This can support coherent integration and a clear contractual counterparty, especially when customers lack internal engineering capacity. The same centralization can reduce visibility into priorities, telemetry, security decisions, or discontinuation. Customers purchase a bundle of functionality, support, and vendor governance rather than source secrecy alone (West, 2003; Wheeler, 2015).

2.2 Economic and Governance Foundations

Software has high development cost and low reproduction cost, making licensing and complementary services central to value capture. Open source firms may earn revenue through support, hosting, certification, integration, dual licensing, managed services, or open-core features. Proprietary firms may use perpetual licenses, subscriptions, usage pricing, maintenance contracts, or platform bundling. The label alone does not reveal incentives or future cost (Fitzgerald, 2006; Riehle, 2012).

Commons theory is relevant because widely used components can generate shared value while maintenance remains underfunded. Beneficiaries may rationally rely on others to contribute, producing a free-rider problem. Foundations, sponsorship, public funding, procurement requirements, and paid maintainer roles can address this imbalance, but each also shapes authority and accountability (Eghbal, 2016; Ostrom, 1990).

Transaction-cost economics explains why organizations sometimes accept lock-in. Integrated proprietary products may reduce search, coordination, and implementation costs in the short term. Switching becomes difficult when data formats, APIs, employee skills, identity systems, workflows, and contracts accumulate around one supplier. Open source can reduce legal restrictions on switching, but migration still requires knowledge, compatible infrastructure, and alternative providers (Shaikh & Cornford, 2011; Williamson, 1985).

Theoretical lens	Explanatory mechanism	LDCF decision criteria

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Theoretical lens	Explanatory mechanism	LDCF decision criteria
Commons governance	Shared infrastructure can be valuable yet underfunded.	Maintainer sustainability; contribution responsibility; concentration risk.
Transaction-cost economics	Coordination, contracting, switching, and asset specificity shape dependency.	TCO; lock-in; governance cost; exit cost.
Open innovation	Firms collaborate around shared complements while competing elsewhere.	Ecosystem participation; supplier plurality; strategic complementarity.
Security engineering	Assurance depends on evidence across actors, artifacts, and operations.	Provenance; dependency control; build integrity; response and maintenance.

3. Methodology / Materials and Methods

3.1 Review design and analytical purpose

This study uses a structured integrative narrative review with interpretive thematic synthesis. The design accommodates heterogeneous empirical studies, conceptual scholarship, legal definitions, standards, and public-policy guidance spanning information systems, economics, security engineering, software governance, public procurement, digital sovereignty, and artificial intelligence. It does not claim exhaustive systematic coverage, statistical representativeness, or a pooled effect. Its purpose is explanatory and framework-building: to identify mechanisms, evidence requirements, tensions, and boundary conditions that explain when alternative licensing and delivery arrangements may create or destroy organizational value.

3.2 Information sources and final search date

The final search update was conducted on September 6, 2026. Sources were located through targeted searches of ACM Digital Library, IEEE Xplore, Google Scholar, publisher platforms indexed by Scopus, and official repositories maintained by the European Commission, CISA, NIST, ENISA, NTIA, and the Open Source Initiative. Backward citation tracing identified foundational work; forward and focused searching prioritized 2019-2026 scholarship on supply-chain security, SBOMs, maintainer sustainability, public-sector development, cloud dependency,

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

digital sovereignty, and AI openness. The search was purposive rather than exhaustive, and database result totals were not treated as a PRISMA flow because the design was not a systematic review.

3.3 Search strategy

Search strings were adapted to each platform's syntax and searched in title, abstract, and keywords where those fields were available. Core query families were: ("open source software" OR proprietary software) AND ("total cost of ownership" OR adoption OR migration); ("open source" OR "closed source") AND (security OR vulnerability OR "software supply chain" OR SBOM); ("vendor lock-in" OR interoperability OR portability) AND (software OR cloud); ("open source governance" OR maintainer sustainability OR project health); and ("digital sovereignty" OR public procurement OR "open source AI" OR "open weights"). Appendix A records platform-specific formulations, fields, filters, and the evidentiary purpose of each search. Because discovery was iterative and included citation chaining, retrieval counts are not presented as evidence of systematic completeness.

3.4 Eligibility and exclusion criteria

Eligible sources made a direct analytical or normative contribution through peer-reviewed empirical evidence, established theory, scholarly synthesis, authoritative licensing definitions, or transparent institutional guidance. Sources had to address at least one decision domain: functional fit, lifecycle cost, security and supply-chain assurance, interoperability and portability, governance and accountability, ecosystem sustainability, strategic autonomy, or exit. Vendor marketing without independently reviewable methods, anonymous comparisons, unsourced market summaries, duplicate claims, and Wikipedia were excluded. Corporate or institutional claims were retained only within the scope their evidence could support.

3.5 Selection process and corpus composition

The sole author conducted relevance screening, full-text eligibility assessment, extraction, appraisal, coding, and synthesis. The revised evidence base contains 40 retained sources: peer-reviewed empirical studies and reviews; theoretical and scholarly books; conference research; and contextual legal, standards, regulatory, and policy documents. Table 2 and Appendix B separate analytical evidence from contextual sources. Sources could inform more than one domain, but each was assigned a principal evidentiary role to prevent institutional guidance or definitions from being misrepresented as comparative empirical proof.

3.6 Data extraction and thematic coding

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

For every retained source, the author recorded year, publication type, software layer, unit of analysis, licensing or delivery model, decision domain, time horizon, principal contribution, evidence role, and principal limitation. Interpretive coding used the domains functional fit; lifecycle cost; security and supply-chain assurance; interoperability and portability; governance and accountability; sustainability; sovereignty; and exit. Evidence labels distinguish empirical findings, theoretical explanations, institutional guidance, legal or definitional material, and author-generated synthesis. Open source rights were separated from capability to exercise them; proprietary service commitments were separated from evidence that makes them credible.

3.7 Critical appraisal and synthesis rules

Each source was appraised at source level using method transparency, direct relevance, comparability of alternatives, lifecycle-boundary completeness, treatment of self-selection or confounding where applicable, and independence from vendor interest. Appendix C reports the matrix and a principal limitation for every source. Vulnerability counts were not interpreted without disclosure, exposure, severity, and remediation context; license price was not treated as total cost; and official guidance was used for standards or policy framing rather than proof of model superiority. The synthesis gives greater interpretive weight to transparent, directly relevant empirical or peer-reviewed scholarship while retaining foundational theory and authoritative definitions for their appropriate roles.

3.8 Single-author procedure and limitations

All stages were performed by one author. Accordingly, Cohen's kappa, Krippendorff's alpha, and percentage agreement are not reported because no second coder existed. The thematic categories and source appraisals are therefore author-generated interpretive judgments, not reliability-tested classifications. Traceability is strengthened through explicit eligibility rules, query families, evidence-role labels, source-characteristics reporting, and source-level appraisal, but these procedures cannot eliminate confirmation bias. This design choice materially limits reproducibility and the strength of comparative claims; the article offers a reasoned decision framework rather than a universal ranking or an exhaustive map of the literature.

4. Results

4.1 Total Cost of Ownership Beyond License Price

Acquisition price is only one component of total cost. A lifecycle assessment should include discovery, procurement, infrastructure, migration, data conversion, integration, customization, training, support, security, compliance, downtime,

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

upgrades, monitoring, and retirement. Open source may remove license fees while increasing internal engineering needs; proprietary software may reduce implementation effort while creating recurring subscriptions and exit costs (Shaikh & Cornford, 2011).

Cost depends on existing capability. An organization with Linux, database, and automation expertise can operate an open stack efficiently, whereas an organization without those skills may need a managed provider. Proprietary software can be cost-effective when standardized workflows and bundled support reduce complexity. Comparisons must use equivalent service levels, reliability targets, and staff assumptions rather than contrasting a self-managed community edition with a fully managed commercial service (Ven et al., 2008; Wheeler, 2015).

Migration and exit are frequently omitted because they occur outside the procurement horizon. Data extraction, format conversion, interface replacement, retraining, parallel operation, and contractual termination can exceed initial implementation cost. A credible business case therefore includes a funded exit scenario and tests whether data, configurations, identities, and automation can be transferred to another solution (Shaikh & Cornford, 2011).

For a defined horizon H, the illustrative lifecycle estimate is: TCO(H) = acquisition + implementation + staffing and operations + support + security and compliance + downtime + change + exit. Organizations should estimate base, favorable, and adverse scenarios, discount future cash flows when material, and document non-monetized risks separately. The expression is a planning template rather than a universal formula; equivalent service levels and workload assumptions are prerequisites for comparison.

Cost component	Measurement basis	Time treatment
Licensing/subscription	Quoted fees, usage tiers, escalation caps	3-5 years
Implementation and migration	Pilot effort, integration, data conversion	Initial plus major upgrades
Staffing and support	Internal FTEs, managed support, training	Annual
Security and compliance	Testing, monitoring, audits, incident duties	Annual and event-driven

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Cost component	Measurement basis	Time treatment
Downtime and performance	Expected outage cost and capacity variance	Scenario-based
Customization and change	Extensions, regression testing, roadmap divergence	Per release/change
Exit	Export, parallel operation, retraining, termination	Funded end-of-horizon scenario

Cost category	Open source exposure	Proprietary exposure	Evidence to request
Acquisition	Usually low or no license fee; enterprise support may be paid.	License, subscription, usage, or bundled fees.	Three-to-five-year price scenarios.
Implementation	Integration and internal expertise may dominate.	Vendor tooling may simplify setup but constrain choices.	Pilot effort and skills inventory.
Operations	Flexible provider choice; organization may own more responsibility.	Contracted support and managed updates; supplier dependency.	Service levels, staffing, and incident history.
Change	Source modification possible, but forks are costly.	Roadmap and customization controlled by vendor.	Upgrade compatibility and customization policy.
Exit	Legal portability can be strong; practical migration still costly.	Formats, APIs, contracts, and bundling can raise switching cost.	Tested export and replacement plan.

4.2 Security: Visibility Is Not Assurance

Open source permits inspection, but visible code is secure only when capable people review, test, and maintain it. Proprietary secrecy can limit independent analysis, but commercial vendors may fund dedicated security teams and coordinated response. Neither 'many eyes' nor 'security through obscurity' is a sufficient assessment. Evidence should address secure development, review

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

coverage, vulnerability handling, release integrity, dependency control, and response performance (Schryen, 2011; Wheeler, 2015).

Modern products contain dependency trees that cross licensing models. A proprietary application may include hundreds of open source packages, while an open source deployment may depend on proprietary cloud, identity, hardware, or support services. Software bills of materials, provenance, signed releases, reproducible builds, dependency inventories, and vulnerability-management processes provide more actionable assurance than the top-level label (CISA, 2023; NIST, 2022).

Supply-chain incidents demonstrate that compromise can enter through maintainers, build systems, package registries, credentials, updates, or dependencies. Concentrated maintainer access can create risk in an open project; compromised vendor signing or update infrastructure can distribute harm at proprietary scale. Security architecture should therefore limit trust, protect release pipelines, verify artifacts, and prepare for dependency failure in both models (CISA, 2023; NIST, 2022).

Disclosure statistics require caution. Open projects may publicly document vulnerabilities that closed suppliers handle less visibly; larger and older projects naturally accumulate more reports. Counts should be normalized by exposure, severity, time to remediation, exploitability, and deployment. Independent testing and operational incident evidence are stronger than claims that one licensing model is inherently safer (Schryen, 2011).

Security layer	Evidence to request
Source code	Review coverage, testing, defect handling
Dependencies	Inventory, vulnerable/transitive packages, replacement
Build and release	Provenance, reproducibility, signing, isolated workers
Update channel	Authentication, rollback, cadence, compromise response
Deployment	Hardening, identity, privilege, configuration drift
Cloud and external services	Jurisdiction, concentration, availability, portability
Incident response	Disclosure, SLA, remediation time, evidence retention
Long-term maintenance	End-of-life, maintainer/supplier continuity, patch backports

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

4.3 Governance, Accountability, and Sustainability

Open source governance ranges from a single maintainer to elected councils, nonprofit foundations, and corporate-led projects. Relevant questions include who can merge code, release versions, change licenses, allocate trademarks, appoint leaders, and resolve disputes. A public repository does not necessarily mean distributed authority. Contributor concentration and employer dependence should be evaluated explicitly (Crowston et al., 2012; O'Mahony, 2007).

Maintainer sustainability is operational risk. Critical software may be supported by a small group facing unpaid workload, support demands, and security pressure. Organizations that depend on such components should contribute funding, engineering time, testing, documentation, or coordinated security resources rather than treating the commons as costless. CISA's roadmap reflects increasing policy recognition that open source security requires ecosystem investment (CISA, 2023; Eghbal, 2016).

Proprietary governance offers contractual accountability but concentrates decision rights. Suppliers can alter prices, licensing, features, data practices, support windows, or product availability. Contracts can mitigate these risks through service levels, audit rights, data-return provisions, security obligations, escrow, transition assistance, and termination rights, although bargaining power differs between customers (Williamson, 1985).

4.4 Interoperability, Lock-In, and Architectural Dependency

Open standards and open source are related but distinct. An open source product can use idiosyncratic formats, while a proprietary product can implement well-documented standards. Portability depends on data schemas, APIs, protocols, identity, automation, and operational knowledge. Procurement should therefore test interoperability directly rather than infer it from licensing (European Commission, 2020; Ghosh, 2005).

Lock-in can arise through technical, economic, contractual, and cognitive mechanisms. Data gravity makes large datasets costly to move; proprietary APIs require application changes; volume discounts make alternatives appear more expensive; and employee familiarity discourages migration. Open source reduces some legal barriers but may create dependency on a distribution, cloud service, specialist consultancy, or dominant corporate sponsor (West, 2003; Williamson, 1985).

Modularity limits the blast radius of dependency. Standard interfaces, portable data, infrastructure as code, automated tests, independent backups, and separation of business logic from vendor-specific services preserve options. Multi-cloud rhetoric without tested portability can create duplicate complexity rather than resilience. Exit

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

capability must be demonstrated through exercises, not assumed from diagrams (NIST, 2022; Shaikh & Cornford, 2011).

4.5 Digital Sovereignty and Public-Sector Value

Digital sovereignty concerns the capacity to make and enforce choices about infrastructure, data, security, and continuity. Open source can support inspection, local adaptation, domestic suppliers, and reduced dependence on one foreign vendor. These benefits require skills, institutions, and sustained funding; merely hosting code locally does not create autonomy if updates and expertise remain externally controlled (European Commission, 2021).

Recent scholarship treats sovereignty as layered and contested rather than as national hosting alone. Cloud infrastructure, standards, data, identity, applications, skills, and regulation may be controlled by different actors. A sovereign outcome therefore requires effective capacity to decide, audit, operate, govern data, negotiate suppliers, and switch when necessary; open licensing can contribute to that capacity but cannot create it without institutions, expertise, funding, and interoperable architecture (Pohle & Thiel, 2020; Sheikh, 2022; Fratini et al., 2024).

Public procurement has additional objectives including transparency, accessibility, long-term records, competition, and avoidance of irreversible dependency. Evaluation should consider open standards, data portability, security maintenance, archival needs, and the ability of multiple suppliers to support the system. Mandating open source without implementation capacity can fail, while procurement written around one proprietary product can exclude alternatives before evidence is considered (European Commission, 2020, 2021).

Local economic value also depends on where expertise and revenue accumulate. Open source may enable regional companies to provide integration and support, but global firms can still dominate hosting and cloud layers. Proprietary procurement may include local partners while core decisions remain abroad. Sovereignty analysis should map control across code, data, infrastructure, skills, trademarks, and governance rather than use national or licensing labels alone (European Commission, 2021).

4.6 Artificial Intelligence and the Limits of “Open”

Artificial intelligence complicates conventional categories. A model may publish weights while withholding training data, code, architecture details, evaluation sets, or usage rights. 'Open weights' can permit local inference and research without satisfying established open source criteria. Precise terminology is necessary

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

because different forms of access support different levels of reproducibility, modification, and accountability (OSI, 2024).

For AI procurement, the LDCF therefore evaluates model access, training-data transparency, license restrictions, fine-tuning rights, local deployment, evaluation evidence, safety duties, compute dependency, update control, and the ability to leave API-based workflows. The OSI definition clarifies that access to weights alone is narrower than full open source AI, while recent security reviews warn that model and data provenance introduce risks not captured by conventional source-code review (Open Source Initiative, 2024; UK Department for Science, Innovation and Technology, 2026).

AI arrangement	What is accessible	Decision implication
Open source software	Code under an approved license	Use, inspect, modify, redistribute
Open-weight model	Model parameters released	May permit local inference or fine-tuning; does not by itself disclose data or training code
Open data	Training/evaluation data and terms documented	Supports scrutiny and reuse subject to lawful rights
Open training code	Pipeline, configuration, and tooling available	Supports examination and partial reproduction
Proprietary API	Remote service under provider control	Convenience and managed operation; dependency on terms, access, updates, and telemetry
Hybrid AI deployment	Open and closed components combined	Requires component-level licenses, provenance, safety evidence, and exit planning

Open models can expand experimentation, audit, language adaptation, and deployment control. They can also shift security, evaluation, hardware, and compliance responsibilities to adopters. Proprietary model services may provide managed infrastructure and rapid improvement but restrict inspection, create usage-based dependency, and change behavior without full customer control. Risk depends on the use case and the evidence available for the specific model and deployment (NIST, 2024; OSI, 2024).

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

The AI comparison should include capability, data governance, privacy, safety evaluation, compute, licensing, support, update control, monitoring, and exit. A lower model-access price may be offset by inference infrastructure and specialist labor, while a convenient API may accumulate switching costs through prompts, tools, embeddings, fine-tuning, and workflow integration. Hybrid strategies can separate sensitive local tasks from externally managed capabilities (NIST, 2024).

4.7 A Contextual Decision Framework

The evidence supports a staged decision rather than a universal score. Organizations should first define the use case, criticality, users, data sensitivity, regulatory duties, service levels, and time horizon. They should then identify realistic candidates and evaluate complete service configurations. Comparing a software package with a hosted platform is invalid unless operational responsibilities are made explicit (Shaikh & Cornford, 2011).

Criterion	Core question	Open-source evidence	Proprietary evidence
Functional fit	Does the complete solution satisfy validated user needs?	Stable releases, roadmap, integrations, accessibility.	Contracted scope, roadmap, configuration, accessibility.
Lifecycle cost	What will ownership, change, and exit cost?	Skills, support, hosting, maintenance, migration.	Fees, escalation, implementation, customization, exit.
Security	Can development and response be verified?	Maintainer health, provenance, review, patch practice.	Secure-development evidence, audit rights, incident duties.
Governance	Who can change the product and terms?	Decision rights, foundation, sponsor concentration.	Vendor authority, contract protections, continuity.
Portability	Can data and workloads move in practice?	Standards, exports, alternative providers, fork viability.	Formats, APIs, transition support, termination rights.
Sovereignty	Where are control and capability located?	Local operation, skills, contributor influence.	Jurisdiction, data control, supplier concentration.

Evidence should be weighted by criticality. A short-lived internal tool may justify rapid adoption and modest exit planning, while identity, health, finance, or public records require stronger continuity, security, and portability. Decision records should

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

state assumptions and uncertainty so that changing prices, governance, or risk can trigger reassessment (NIST, 2022).

5. Discussion

5.1 Why Universal Comparisons Fail

Statements that open source is cheaper, safer, or more innovative are incomplete without specifying the product, adopter, and timeframe. The same is true of claims that proprietary software is easier, more accountable, or professionally supported. Mature open projects can outperform weak vendors, and well-governed proprietary products can outperform abandoned repositories. Licensing changes the option set; organizational and ecosystem conditions determine whether options become outcomes (Schryen, 2011; Ven et al., 2008).

Software layers also distribute value differently. Open source is prominent in operating systems, runtimes, databases, libraries, and orchestration, while proprietary control may concentrate in hosted interfaces, data, marketplaces, management planes, and customer relationships. Calling an entire service open or closed can hide this layered allocation of control (Fitzgerald, 2006; West, 2003).

5.2 Hybrid Portfolios as the Normal Case

Most organizations operate hybrid portfolios because no single governance model satisfies every need. A proprietary application may run on open infrastructure; an open application may depend on proprietary hosting; a firm may self-host critical workloads and purchase managed services elsewhere. Hybrid design should be intentional, with boundaries selected according to sensitivity, capability, differentiation, and exit risk (Ven et al., 2008).

Hybridization does not automatically reduce dependency. Combining many products can increase integration burden, fragmented accountability, and licensing complexity. Architecture inventories should map components, licenses, maintainers, vendors, data flows, and support relationships. This makes concentration visible even when procurement appears diversified (CISA, 2023; NIST, 2022).

5.3 Policy and Management Implications

Organizations should establish an open source program or equivalent governance function when dependency becomes material. Responsibilities include component inventory, license compliance, contribution policy, vulnerability response, upstream engagement, and project-health assessment. The goal is not to promote open source indiscriminately but to manage it as strategic infrastructure (CISA, 2023).

Procurement for proprietary services should give equal attention to dependency. Security questionnaires and price comparisons are insufficient without export testing, interface documentation, change notification, auditability, subcontractor

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

visibility, and termination assistance. Negotiating exit only after a dispute or discontinuation is predictably expensive (Shaikh & Cornford, 2011; Williamson, 1985).

Policy should support the production of shared digital infrastructure through funding, procurement, standards, education, and coordinated security. At the same time, public support should preserve plural governance and avoid replacing vendor concentration with state or corporate capture. Transparent criteria and measurable outcomes are preferable to ideological mandates (European Commission, 2021; Ostrom, 1990).

5.4 Research Priorities

Future research should compare complete deployments longitudinally, including migration, staffing, security, upgrades, performance, user outcomes, and exit. Natural experiments around license changes, vendor acquisitions, project forks, and public procurement could reveal how governance affects cost and continuity. Studies should disclose funding and avoid treating vulnerability counts or license prices as complete outcomes (Schryen, 2011; Shaikh & Cornford, 2011).

Research on the Global South should examine whether open source builds local capability or merely relocates dependency to foreign clouds and upstream projects. Relevant measures include local employment, supplier diversity, language support, public-sector capacity, contribution authority, and data control. Sovereignty is better measured through effective capability than through software origin alone (European Commission, 2021).

5.5 Licensing Compatibility and the Risk of Strategic Relicensing

Open source is not a single license. Permissive licenses generally permit broad reuse with limited conditions, while reciprocal or copyleft licenses require specified forms of source availability when covered works are distributed. Network-oriented licenses may extend obligations to software provided as a service, and source-available licenses can expose code while imposing restrictions incompatible with the Open Source Definition. Legal review must therefore identify the exact license, version, dependencies, distribution model, modifications, and intended use (OSI, n.d.; Rosen, 2004).

License compatibility becomes difficult when a product combines components under different obligations. A dependency can require preservation of notices, availability of corresponding source, disclosure of modifications, or limitations on trademark use. Automated scanners support inventories but do not interpret every legal relationship correctly. Organizations need documented review, attribution processes, policies for contribution and modification, and escalation when

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

obligations are unclear. Ignoring compliance can create remediation cost and delay releases even when the underlying software is technically sound (Rosen, 2004).

Strategic relicensing illustrates the difference between access to a current version and durable governance. A copyright holder with sufficient control may move future releases from an approved open source license to a more restrictive source-available model. Existing versions remain under their original licenses, but users must decide whether to remain, fork, migrate, or accept new terms. The practical viability of a fork depends on maintainers, trademarks, community coordination, infrastructure, and customer adoption rather than legal permission alone (Riehle, 2012; West, 2003).

Proprietary agreements also change. Vendors may revise prices, product bundles, support periods, minimum commitments, or cloud terms. The comparable control is not a promise that terms will remain fixed, but an architecture and contract that preserve time to respond. Notification periods, renewal caps, source or data escrow where appropriate, transition support, standards-based interfaces, and tested exports reduce exposure. Relicensing and repricing should therefore be analyzed as related forms of supplier-governance risk (Shaikh & Cornford, 2011; Williamson, 1985).

5.6 Project Health and Supplier Viability

Adoption reviews often count contributors, commits, or releases as simple measures of project health. These indicators can mislead. High commit volume may reflect automated changes, while a stable mature component may need few releases. More informative evidence includes maintainer concentration, review delay, unresolved security issues, release predictability, contributor retention, funding diversity, documentation, governance clarity, dependency freshness, and responsiveness to users. Measures should be interpreted against project size and criticality (Crowston et al., 2012; Eghbal, 2016).

The bus factor - the number of people whose absence would jeopardize continuity - is especially important for critical dependencies. A widely downloaded library can depend on one or two maintainers, creating an imbalance between social value and available labor. Downstream organizations should identify such concentration and respond through funding, upstream contributions, internal expertise, maintained mirrors, or replacement planning. Popularity is evidence of exposure, not proof of sustainability (CISA, 2023; Eghbal, 2016).

Vendor viability requires parallel scrutiny. Revenue growth or market recognition does not guarantee product continuity, and acquisition can alter priorities, prices, or support. Customers should examine financial resilience, ownership, product concentration, subcontractors, incident history, roadmap credibility, customer references, and contractual transition obligations. A proprietary supplier may offer

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

stronger immediate support than a volunteer project yet create a larger discontinuity if the entire service is withdrawn (Williamson, 1985).
Foundations can reduce dependence on a single company by holding trademarks, funds, infrastructure, or governance responsibilities, but institutional form alone is not decisive. Boards may be dominated by major sponsors, contributors may lack meaningful representation, and technical authority may remain concentrated. Governance assessment should compare formal rules with actual decision patterns and contribution data (O'Mahony, 2007; West & Gallagher, 2006).

5.7 Comparative Scenarios and Boundary Conditions

For standardized infrastructure, a mature open source component may offer strong portability and a competitive support market, particularly when the organization possesses engineering capability. The benefit weakens when customization creates a private fork that cannot absorb upstream updates. A disciplined policy favors configuration, modular extensions, and upstream contribution before divergence. Proprietary infrastructure may remain appropriate when a managed service materially reduces operational risk and exit requirements are satisfied (Ven et al., 2008; West, 2003).
For end-user productivity software, workflow fit, accessibility, file compatibility, training, and collaboration networks may outweigh source availability. Switching costs can arise because partners exchange complex documents or employees depend on specialized functions. Open alternatives should be evaluated through representative tasks rather than feature lists, while proprietary incumbents should be tested for export fidelity and long-term format access. The relevant outcome is successful work and preserved information, not ideological consistency (Shaikh & Cornford, 2011).
For safety-critical or regulated systems, source access may support audit and long-term maintenance, but certification, validated configurations, controlled change, traceability, and accountable support are equally important. Unreviewed local modification can invalidate assurance, while a closed component without sufficient evidence may create unacceptable opacity. Procurement should specify assurance artifacts, update duties, vulnerability response, retention, and continuity independently of license category (NIST, 2022; Wheeler, 2015).
For education and research, open source can expand access, reproducibility, and the ability to study implementation. Nevertheless, learners and institutions still require documentation, accessible interfaces, hardware, connectivity, and support. A nominally free tool that consumes extensive staff time or excludes users with disabilities can fail educational goals. Evaluation should include pedagogical outcomes and support capacity as well as acquisition price (European Commission, 2021).

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

These scenarios demonstrate a consistent boundary condition: open source creates option value when an organization can exercise the rights it receives, while proprietary software creates service value when contractual and technical commitments are credible. Rights without capability may remain theoretical; service without exit may become dependency. Balanced assessment therefore measures both present performance and future freedom of action (Shaikh & Cornford, 2011; West & Gallagher, 2006).

Scenario	Potentially stronger open-source condition	Potentially stronger proprietary condition	Non-negotiable check
Cloud infrastructure	Strong internal platform team and standards-based architecture.	Managed service materially reduces operational burden.	Workload and data exit tested.
Office/productivity	Open formats and representative workflows perform reliably.	Compatibility network and support outweigh license flexibility.	Accessibility and export fidelity.
Regulated system	Auditable code plus controlled, validated maintenance.	Vendor supplies credible assurance and accountability.	Evidence, traceability, continuity.
Education/research	Reproducibility and local adaptation are central.	Specialized support or content is indispensable.	Whole-life cost and learner access.
Critical dependency	Healthy governance, funded maintainers, provider plurality.	Supplier is viable with strong contractual protection.	Fallback, monitoring, and replacement plan.

5.8 Lifecycle Monitoring and Reassessment

The selection decision should be revisited because products and dependencies evolve after procurement. Open projects can lose maintainers, change governance, split into competing forks, or accumulate vulnerable dependencies. Proprietary suppliers can change ownership, licensing, hosting regions, support windows, or strategic priorities. A product that was appropriate at adoption may become unsuitable without any failure in its original evaluation. Periodic review should therefore monitor technical, economic, contractual, and governance indicators (CISA, 2023; NIST, 2022).

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Monitoring should include release and patch cadence, end-of-life notices, unresolved severe vulnerabilities, dependency freshness, maintainer or supplier concentration, price changes, service performance, portability tests, support responsiveness, and internal skill coverage. Thresholds need named owners and predefined responses. For example, loss of security maintenance may trigger compensating controls and replacement analysis, while a major pricing change may trigger an updated total-cost comparison (Shaikh & Cornford, 2011).

Organizations should also distinguish ordinary updates from material changes. A new major version, license change, acquisition, cloud-only transition, incompatible file format, or altered telemetry practice may require security, legal, privacy, accessibility, and architectural reassessment. Automatic renewal can conceal such changes when procurement focuses only on price. A structured change register preserves the rationale for acceptance and identifies assumptions that no longer hold (NIST, 2022; Williamson, 1985).

Exit exercises provide the strongest evidence that alternatives remain viable. A limited exercise can export representative data, restore it in an independent environment, rebuild essential configuration, verify permissions, and measure time and loss of fidelity. The exercise may reveal undocumented APIs, proprietary extensions, unavailable knowledge, or dependencies on one consultant. Findings should update both risk estimates and contractual negotiations. Exit planning is thus not preparation for inevitable abandonment; it is a mechanism for maintaining credible choice throughout the lifecycle (Shaikh & Cornford, 2011).

5.9 Limitations of This Review

This integrative narrative review is not systematic and does not estimate pooled effects. Methodologically, its purposive discovery process did not preserve exhaustive database retrieval counts, and screening, extraction, appraisal, and interpretive coding were conducted by one author without inter-rater reliability testing. Evidentially, the corpus remains heterogeneous, direct head-to-head comparisons are scarce, and institutional guidance cannot substitute for empirical outcome research. Conceptually, open and proprietary categories contain diverse licenses, governance arrangements, cloud dependencies, and hybrid business models. Temporally, projects, suppliers, licenses, regulations, AI models, and security practices change rapidly. External validity also depends on adopter capability, criticality, jurisdiction, architecture, contract design, and time horizon. The LDCF is therefore intended to structure organizational evaluation, not to generate a universal numerical ranking between open source and proprietary software.

6. Conclusion

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Open source and proprietary software allocate permissions, control, responsibility, and dependency differently. Open source can improve inspectability, adaptability, supplier plurality, and local control, but requires sustainable maintenance and organizational capability. Proprietary software can provide integrated service and contractual accountability, but may create opacity, unilateral change, and costly exit. Neither license model substitutes for evidence about the actual system (CISA, 2023; Schryen, 2011).

A defensible choice begins with functional requirements and extends through total cost, security, interoperability, governance, sustainability, sovereignty, and exit. These criteria must be evaluated at the level of the complete deployment, including cloud, data, identities, integrations, staff, and contracts. Hybrid portfolios are often appropriate, provided their dependencies are visible and governed (Shaikh & Cornford, 2011; Ven et al., 2008).

The central question is not whether software is open or proprietary in the abstract, but who can inspect, change, operate, secure, support, and replace it under realistic conditions. Organizations that preserve tested options and invest in the ecosystems on which they depend are better positioned to obtain value from either model (European Commission, 2021; Ostrom, 1990).

Appendix A Search Strategy

The search was a transparent, purposive component of an integrative narrative review. It was not registered as a systematic review, and exact retrieval totals were not preserved. The following log reports the reproducible query families and their purposes without implying PRISMA-style completeness.

Platform	Fields and filters	Representative syntax	Purpose
ACM Digital Library	Title/abstract; 2019-2026; research articles and proceedings	("open source" AND (governance OR sustainability OR maintainers)) OR ("software supply chain" AND security)	Governance, sustainability, supply chain
IEEE Xplore	Metadata/abstract; 2019-2026; journals and conferences	("open source software" OR proprietary) AND (security OR SBOM OR adoption OR lock-in)	Security, adoption, SBOM, lock-in
Google	All fields; first 200	"digital sovereignty"	Sovereignty,

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Platform	Fields and filters	Representative syntax	Purpose
Scholar	relevance-ranked results per focused query; 2019-2026 plus seminal tracing	software; "cloud vendor lock-in" portability; "open weights" "open source AI"	cloud, AI
Scopus-indexed publisher platforms	Title/abstract/keywords where available; 2019-2026	(open source OR proprietary) AND (TCO OR procurement OR interoperability OR sustainability)	Recent peer-reviewed evidence
Official repositories	Current documents; issuing bodies only	site-specific searches for SSDF, SBOM, open source security, AI openness, and sovereignty	Standards, definitions, regulatory context

Appendix B Source Characteristics

Source	Type	Context	Principal contribution	Role
Alami et al., 2024	Empirical	OSS communities	Sustainability metrics and software quality	Analytical
CISA, 2023	Policy guidance	Software ecosystem	Security roadmap and stewardship	Contextual
Crowston et al., 2012	Review	OSS development	Governance and participation evidence	Analytical
Eghbal, 2016	Research report	Digital infrastructure	Maintainer labor and underfunding	Analytical
European Commission, 2020	Policy	Public administration	Open source strategy	Contextual
European Commission,	Policy study	EU economy	Independence and innovation	Contextual

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Type	Context	Principal contribution	Role
2021				
Fitzgerald, 2006	Conceptual	Commercial OSS	Hybrid production transformation	Analytical
Fratini et al., 2024	Systematic review	Digital sovereignty	Four governance models	Analytical
Ghosh, 2005	Policy report	Interoperability	Open standards distinction	Contextual
Lerner & Tirole, 2002	Theory	OSS economics	Contributor and firm incentives	Analytical
Linåker et al., 2024	Qualitative study	Maintenance labor	Human infrastructure and support	Analytical
Linåker et al., 2025	Mixed methods	Public-sector OSS	Formal organization and procurement	Analytical
Munir & Mols, 2021	Practice study	Open source program offices	Organizational governance capability	Analytical
NIST, 2022	Standard/guidance	Secure development	SSDF practices	Contextual
NIST, 2024	Guidance	Generative AI	AI risk governance	Contextual
O'Mahony, 2007	Empirical/conceptual	OSS governance	Community management and authority	Analytical
Okafor et al., 2022	Systematization	Software supply chain	Transparency, validity, separation	Analytical
Open Source	Definition	Software licensing	Open Source Definition	Definitional

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Type	Context	Principal contribution	Role
Initiative, n.d.				
Open Source Initiative, 2024	Definition	Artificial intelligence	Open Source AI Definition	Definitional
Ostrom, 1990	Theory	Common-pool resources	Commons governance	Analytical
Pohle & Thiel, 2020	Conceptual review	Digital sovereignty	Concept genealogy and tensions	Analytical
Prana et al., 2021	Empirical	OSS dependencies	Response to vulnerable dependencies	Analytical
Raymond, 1999	Foundational essay	OSS culture	Distributed development model	Contextual
Riehle, 2012	Conceptual	Commercial OSS	Single-vendor business model	Analytical
Robles et al., 2019	Review/commentary	OSS evolution	Mainstream institutionalization	Analytical
Rosen, 2004	Legal scholarship	Licensing	Compatibility and obligations	Analytical
Schryen, 2011	Review	Software security	Limits of open/closed comparison	Analytical
Shaikh & Cornford, 2011	Policy research	TCO	Lifecycle cost and adoption	Analytical

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Type	Context	Principal contribution	Role
Sheikh, 2022	Conceptual	European sovereignty	Layered technology stack	Analytical
Stalnaker et al., 2024	Empirical study	SBOMs	Production and consumption practices	Analytical
Ven et al., 2008	Decision article	OSS adoption	Organizational adoption conditions	Analytical
West, 2003	Conceptual	Openness strategies	Degrees of openness	Analytical
West & Gallagher, 2006	Conceptual/cases	Open innovation	Firm participation paradox	Analytical
Wheeler, 2015	Technical guidance	Secure programming	Engineering practices	Contextual
Williamson, 1985	Theory	Organizations/contracts	Transaction and switching costs	Analytical
ENISA, 2021	Threat report	Supply chains	Attack taxonomy and controls	Contextual
NTIA, 2021	Minimum elements	SBOM	Baseline transparency fields	Contextual
NTIA, 2024	Policy report	Open-weight AI	Benefits and risk policy	Contextual
UK DSIT, 2026	Evidence review	OSS and open AI	Distinct security profiles	Contextual
Hoffmann et al., 2024	Economic working paper	OSS economy	Value of widely used OSS	Analytical

Appendix C Critical Appraisal Matrix

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Method transparency	Independence	Review relevance	Principal limitation
Alami et al., 2024	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
CISA, 2023	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
Crowston et al., 2012	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Eghbal, 2016	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
European Commission, 2020	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
European Commission, 2021	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
Fitzgerald, 2006	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Fratini et al., 2024	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Ghosh, 2005	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Method transparency	Independence	Review relevance	Principal limitation
				causal comparison.
Lerner & Tirole, 2002	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Linåker et al., 2024	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Linåker et al., 2025	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Munir & Mols, 2021	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
NIST, 2022	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
NIST, 2024	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
O'Mahony, 2007	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Okafor et al., 2022	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Method transparency	Independence	Review relevance	Principal limitation
Open Source Initiative, n.d.	High for definition	Standards body	Direct for terminology	Does not establish comparative outcomes.
Open Source Initiative, 2024	High for definition	Standards body	Direct for terminology	Does not establish comparative outcomes.
Ostrom, 1990	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Pohle & Thiel, 2020	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Prana et al., 2021	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Raymond, 1999	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
Riehle, 2012	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Robles et al., 2019	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Rosen, 2004	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Method transparency	Independence	Review relevance	Principal limitation
				limits generalization.
Schryen, 2011	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Shaikh & Cornford, 2011	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Sheikh, 2022	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Stalnaker et al., 2024	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Ven et al., 2008	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
West, 2003	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
West & Gallagher, 2006	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
Wheeler,	Moderate-	Institutional	Contextual	Guidance or policy

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Source	Method transparency	Independence	Review relevance	Principal limitation
2015	high	perspective		evidence is not causal comparison.
Williamson, 1985	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.
ENISA, 2021	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
NTIA, 2021	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
NTIA, 2024	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
UK DSIT, 2026	Moderate-high	Institutional perspective	Contextual	Guidance or policy evidence is not causal comparison.
Hoffmann et al., 2024	Moderate-high	Generally independent	Direct or supporting	Context, method, or historical period limits generalization.

Ethical Approval
Ethical approval was not required because this study analyzed publicly available literature and institutional documents and involved no human participants, identifiable personal data, animals, or intervention.

Funding
The author received no financial support for the research, authorship, or publication of this article.

Conflicts of Interest
The author declares no conflict of interest.

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

References

Alami, A., Pardo, R., & Linåker, J. (2024). Free open source communities sustainability: Does it make a difference in software quality? Empirical Software Engineering, 29, Article 114. https://doi.org/10.1007/s10664-024-10529-6

Crowston, K., Wei, K., Howison, J., & Wiggins, A. (2012). Free/libre open-source software development: What we know and what we do not know. ACM Computing Surveys, 44(2), Article 7. https://doi.org/10.1145/2089125.2089127

Cybersecurity and Infrastructure Security Agency. (2023). Open source software security roadmap. https://www.cisa.gov/resources-tools/resources/cisa-open-source-software-security-roadmap

Eghbal, N. (2016). Roads and bridges: The unseen labor behind our digital infrastructure. Ford Foundation.

European Commission. (2020). Open source software strategy 2020-2023. https://commission.europa.eu/about/departments-and-executive-agencies/digital-services/open-source-software-strategy_en

European Commission. (2021). Study about the impact of open source software and hardware on technological independence, competitiveness and innovation in the EU economy. https://digital-strategy.ec.europa.eu/en/library/study-about-impact-open-source-software-and-hardware-technological-independence-competitiveness-and

European Union Agency for Cybersecurity. (2021). ENISA threat landscape for supply chain attacks. https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks

Fitzgerald, B. (2006). The transformation of open source software. MIS Quarterly, 30(3), 587-598. https://doi.org/10.2307/25148740

Fratini, S., Hine, E., Novelli, C., Roberts, H., & Floridi, L. (2024). Digital sovereignty: A descriptive analysis and a critical evaluation of existing models. Digital Society, 3, Article 59. https://doi.org/10.1007/s44206-024-00146-7

Ghosh, R. A. (2005). Open standards and interoperability report. European Commission.

Hoffmann, M., Nagle, F., & Zhou, Y. (2024). The value of open source software. Harvard Business School Working Paper No. 24-038.

Lerner, J., & Tirole, J. (2002). Some simple economics of open source. Journal of Industrial Economics, 50(2), 197-234. https://doi.org/10.1111/1467-6451.00174

Linåker, J., Butler, S., Brax, C., Hall, D., & Papatheocharous, E. (2024). Sustaining maintenance labor for healthy open source software projects through human infrastructure. Proceedings of the 2024 IEEE/ACM 17th International

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

Conference on Cooperative and Human Aspects of Software Engineering. https://doi.org/10.1145/3674805.3686667

Linåker, J., Lundell, B., Servant, F., Gamalielsson, J., Muto, S., & Robles, G. (2025). Public sector open source software projects: How is development organized? Empirical Software Engineering, 30, Article 80. https://doi.org/10.1007/s10664-025-10626-0

Munir, H., & Mols, C. E. (2021). The rise of open source program offices. IT Professional, 23(1), 27-33.

National Institute of Standards and Technology. (2022). Secure software development framework (SSDF) version 1.1. https://doi.org/10.6028/NIST.SP.800-218

National Institute of Standards and Technology. (2024). Artificial intelligence risk management framework: Generative artificial intelligence profile. https://doi.org/10.6028/NIST.AI.600-1

National Telecommunications and Information Administration. (2021). The minimum elements for a software bill of materials. U.S. Department of Commerce.

National Telecommunications and Information Administration. (2024). Dual-use foundation models with widely available model weights. U.S. Department of Commerce.

O'Mahony, S. (2007). The governance of open source initiatives: What does it mean to be community managed? Journal of Management & Governance, 11, 139-150. https://doi.org/10.1007/s10997-007-9024-7

Okafor, C., Schorlemmer, T. R., Torres-Arias, S., & Davis, J. C. (2022). SoK: Analysis of software supply chain security by establishing secure design properties. Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses, 15-24. https://doi.org/10.1145/3560835.3564556

Open Source Initiative. (2024). The open source AI definition, version 1.0. https://opensource.org/ai/open-source-ai-definition

Open Source Initiative. (n.d.). The open source definition. https://opensource.org/osd

Ostrom, E. (1990). Governing the commons. Cambridge University Press.

Pohle, J., & Thiel, T. (2020). Digital sovereignty. Internet Policy Review, 9(4). https://doi.org/10.14763/2020.4.1532

Prana, G. A. A., Sharma, A., Shar, L. K., Foo, D., Santosa, A. E., Sharma, A., & Lo, D. (2021). Out of sight, out of mind? How vulnerable dependencies affect open-

CC BY

Nexus Science Review
INTERNATIONAL JOURNAL OF INTERDISCIPLINARY RESEARCH

Received: 10/08/2026 - **Accepted:** 07/08/2026
Vol: 01.04
DOI: 10.66104/9x7pbx54
Pages: 1-32

source projects. Empirical Software Engineering, 26, Article 59. https://doi.org/10.1007/s10664-021-09959-3

Raymond, E. S. (1999). The cathedral and the bazaar. O'Reilly Media.

Riehle, D. (2012). The single-vendor commercial open source business model. Information Systems and e-Business Management, 10, 5-17. https://doi.org/10.1007/s10257-010-0149-x

Robles, G., Steinmacher, I., Adams, P., & Treude, C. (2019). Twenty years of open source software: From skepticism to mainstream. IEEE Software, 36(6), 12-15.

Rosen, L. (2004). Open source licensing: Software freedom and intellectual property law. Prentice Hall.

Schryen, G. (2011). Is open source security a myth? Communications of the ACM, 54(5), 130-140. https://doi.org/10.1145/1941487.1941516

Shaikh, M., & Cornford, T. (2011). Total cost of ownership of open source software. UK Cabinet Office.

Sheikh, H. (2022). European digital sovereignty: A layered approach. Digital Society, 1, Article 25. https://doi.org/10.1007/s44206-022-00025-z

Stalnaker, T., Wintersgill, N., Chaparro, O., Di Penta, M., & German, D. M. (2024). A comprehensive study of bills of materials for software systems. Proceedings of the 46th IEEE/ACM International Conference on Software Engineering.

UK Department for Science, Innovation and Technology. (2026). A study of cybersecurity literature on open-source software and AI. UK Government.

Ven, K., Verelst, J., & Mannaert, H. (2008). Should you adopt open source software? IEEE Software, 25(3), 54-59. https://doi.org/10.1109/MS.2008.73

West, J. (2003). How open is open enough? Research Policy, 32(7), 1259-1285. https://doi.org/10.1016/S0048-7333(03)00052-0

West, J., & Gallagher, S. (2006). Challenges of open innovation: The paradox of firm investment in open-source software. R&D Management, 36(3), 319-331. https://doi.org/10.1111/j.1467-9310.2006.00436.x

Wheeler, D. A. (2015). Secure programming HOWTO. Linux Foundation.

Williamson, O. E. (1985). The economic institutions of capitalism. Free Press.

CC BY