# Decision Debt in AI-Generated Work

Preserving Observability, Traceability, Modularity, and Changeability

Johnny Kao

Independent Researcher, Tokyo, Japan

ORCID: 0009-0004-1446-6001

*Working Paper*

## Abstract

Artificial intelligence can produce work that is functionally correct at delivery yet difficult to inspect, reconstruct, or modify later. This paper develops the concept of decision debt for AI-generated work: the future cost created when an acceptable output is retained without preserving the assumptions, evidence, dependencies, and interfaces needed to understand and safely change it. The concept extends but does not replace technical debt. Whereas technical debt traditionally concerns implementation and architectural compromises that increase future maintenance cost, decision debt focuses on lost decision structure across software, analytical work, and agentic workflows. The paper proposes four preservation properties - observability, traceability, modularity, and changeability - and a future-changeability test for evaluating whether AI-generated artifacts remain governable after delivery. It also offers a qualitative decision-debt checklist and a risk-adjusted preservation heuristic for deciding how much reconstruction evidence to retain. It also distinguishes external reconstruction evidence from private model reasoning: maintainability does not require access to an AI system's hidden chain of thought, but it does require sufficient records of sources, actions, tests, assumptions, and artifacts to reconstruct consequential decisions. The contribution is integrative rather than a claim that provenance, maintainability, or technical debt are individually new. The central claim is that correctness is a present-tense property, while changeability is a future-tense property; reliable AI-generated work requires both.

Keywords: AI-generated work; decision debt; maintainability; observability; traceability; provenance; changeability; agentic AI

*Author note: This working paper develops and formalizes the decision-debt framework first introduced in Kao (2026d).*

## 1. Introduction

Generative and agentic AI systems can now produce software, analyses, reports, plans, and operational changes at a speed that makes output quality the obvious first concern. Does the code compile? Does the answer look correct? Did the agent complete the requested task? These questions matter, but they are present-tense questions. Long-lived work creates a second problem that may appear only after delivery: can a future person or future AI understand enough of the artifact to change it safely?

The distinction is visible in software engineering. A generated feature can pass its tests and still embed duplicated rules, undocumented assumptions, hidden dependencies, or interfaces that are difficult to alter. Recent empirical work confirms that the quality profile of AI-generated code cannot be reduced to immediate functional correctness. Large-scale comparisons report distinct defect and vulnerability patterns in AI-authored code, while studies of generated patches and maintenance find that maintainability problems, new defects, and continued human intervention remain material concerns (Cotroneo et al., 2025; Nunes et al., 2025; Sawada et al., 2026; Sun et al., 2026).

The same problem extends beyond code. An AI-generated market analysis may reach a reasonable recommendation while failing to preserve which evidence supports which claim. A research agent may collect sources without recording which assumptions shaped selection. An operations agent may reach an acceptable end state while leaving insufficient records of the actions, exceptions, and policy decisions that produced it. When future conditions change, the artifact may be difficult to revise even though the original output was acceptable.

This paper develops the concept of decision debt for this broader class of failures. Decision debt is the future cost created when AI-generated work is accepted without preserving the assumptions, evidence, dependencies, and interfaces needed to reconstruct and safely modify that work later. The concept is related to technical debt but is deliberately narrower in one dimension and broader in another: it focuses on lost decision structure rather than all forms of implementation debt, and it applies to software, analytical artifacts, and agentic workflows rather than code alone.

The contribution is integrative. Technical debt, provenance, observability, traceability, reproducibility, and maintainability are established concepts. The proposed framework connects them around a future-changeability problem created by AI-generated work. It identifies four preservation properties - observability, traceability, modularity, and changeability - and argues that reliable maintenance depends on external reconstruction evidence rather than access to a model's private reasoning. The central claim is simple: correctness is a present-tense property; changeability is a future-tense property. A serious AI workflow needs both.

# 2. From Technical Debt to Decision Debt

## 2.1 Technical debt and future maintenance cost

Technical debt is a useful starting point because it reframes a successful present-state system as potentially expensive to change later. Kruchten, Nord, and Ozkaya (2012) describe technical debt as a way to reason about development decisions and their consequences for future system evolution. Sculley et al. (2015) extended the logic to machine-learning systems, showing how quick wins can create hidden maintenance costs through entanglement, boundary erosion, undeclared consumers, data dependencies, configuration issues, and other system-level effects.

AI-assisted development intensifies the relevance of this perspective because generation cost can fall faster than review and maintenance capacity. Recent evidence is mixed in ways that are important for this paper. Molison et al. (2025) find that LLM-generated code can perform well on some maintainability and reliability measures, while more complex tasks can still introduce structural issues. Cotroneo et al. (2025), in a study of more than 500,000 code samples, identify different defect profiles between AI- and human-authored code rather than a simple universal quality ordering. Nunes et al. (2025) find that LLM attempts to repair maintainability issues frequently introduce compilation errors, test failures, or new maintainability problems. Sun et al. (2026) similarly argue that generated code must be evaluated beyond functional correctness and report practitioner concern that AI generation can accelerate technical debt.

The important point is therefore not that AI-generated code is inherently less maintainable. It is that fast functional success does not settle the question of future maintainability. The same output can be correct now and expensive to reinterpret or modify later.

## 2.2 Decision debt as a distinct analytical lens

Decision debt focuses on a specific mechanism within this broader maintenance problem: the loss of preserved decision structure. It arises when an artifact survives but the assumptions, evidence, dependencies, or interfaces needed to understand consequential choices do not. This can happen even when the implementation itself is clean and even when the initial result is correct.

Consider two travel plans that both satisfy the same budget and arrival deadline. One uses a direct flight; another uses a bus, rental car, and train. If only the final success condition is preserved, the plans look equivalent. If the traveler later develops a mobility constraint, the internal structure of the chosen plan suddenly matters. The future problem is not simply whether the original plan worked. It is whether the decision can be reconstructed and changed without starting again from zero.

**Table 1 positions decision debt relative to technical debt. The distinction is analytical rather than absolute; the two can coexist and often reinforce each other.**

**Table 1. Technical debt and decision debt in AI-generated work**

| Dimension | Technical debt | Decision debt |
|---|---|---|
| Primary concern | Implementation or architectural choices that increase future maintenance cost. | Loss of assumptions, evidence, dependencies, or interfaces needed to reconstruct and safely revise a decision or artifact. |
| Typical locus | Code, architecture, data pipelines, configurations, system dependencies. | Code plus analyses, plans, research artifacts, agent actions, acceptance decisions, and other AI-generated work. |
| Present-state appearance | System may work despite structural compromises. | Output may be acceptable despite missing reconstruction evidence. |
| Future failure mode | Changes become expensive, risky, or slow. | Future actors cannot determine what must change, why a result was produced, or which dependencies will be affected. |
| Relationship | Can create decision debt when important design choices become opaque. | Can exist without obvious code debt, and can amplify technical debt when hidden assumptions are embedded in implementation. |

## 3. The Future-Changeability Gap

Many AI evaluations are optimized for the moment of completion. Software benchmarks ask whether generated code solves a task or passes tests. Analytical workflows ask whether a report is accurate enough to use. Agent benchmarks often score task success, tool use, latency, or cost. These measures are necessary, but they do not fully capture whether the resulting work remains intelligible and modifiable after the environment changes.

Recent software research is beginning to expose this gap. Sun et al. (2026) explicitly distinguish functional correctness from non-functional quality characteristics and show that improvements in one quality dimension can trade off against others. Sawada et al. (2026), studying more than 1,000 files from AI-generated pull requests, find that human developers perform the large majority of subsequent maintenance, indicating that autonomous generation does not eliminate the need for future human intervention. These findings are specific to software, but the broader implication is general: delivery is not the end of the artifact's lifecycle.

Provenance research addresses a complementary part of the problem. W3C PROV models the entities, activities, and agents involved in producing an artifact, supporting uses such as trust assessment, compliance checking, and reproduction (W3C, 2013). Schlegel and Sattler (2025) show that even modern ML artifact systems often capture incomplete or coarse-grained provenance, motivating end-to-end models that connect version-control and ML-pipeline artifacts. Souza et al. (2025) extend provenance to agentic workflows through PROV-AGENT, linking prompts, responses, decisions, workflow context, and downstream effects. These approaches strengthen observability and traceability, but provenance alone does not guarantee that an artifact is modular or safe to change.

The future-changeability gap therefore sits between correctness and provenance. A trace can tell us what happened without ensuring that the resulting structure is easy to modify. A modular artifact can be easy to change while lacking enough provenance to understand why the change is appropriate. Decision debt treats these properties as complementary rather than interchangeable.

## 4. Definition of Decision Debt

Definition 1 (Decision debt). Decision debt is the future cost created when AI-generated work is accepted without preserving the assumptions, evidence, dependencies, and interfaces needed to reconstruct and safely modify that work later.

This definition contains four boundaries. First, decision debt is future-oriented: the debt may not be visible at delivery. Second, it can arise from successful outputs; immediate error is not required. Third, it concerns preserved structure rather than access to every internal model step. Fourth, it is consequential only when

future change, review, or accountability matters. Ephemeral low-stakes outputs may rationally preserve very little.

The concept also separates decision debt from ordinary documentation debt. Missing documentation can contribute to decision debt, but the relevant question is functional: can a future actor reconstruct enough of the consequential structure to intervene intelligently? A perfectly documented transcript that does not connect evidence to decisions can still leave high decision debt. Conversely, a compact set of sources, diffs, tests, assumptions, and interface contracts may preserve what matters without recording every intermediate step.

# 5. Four Preservation Properties

The framework proposes four properties that reduce decision debt: observability, traceability, modularity, and changeability. They are not presented as new concepts individually. Their value here is as a compact set of preservation requirements for AI-generated work.

**Table 2. Four preservation properties**

| Property | Operational question | Characteristic failure when weak |
|---|---|---|
| Observability | Can we tell what the system actually did? | Actions, tool calls, changes, or side effects cannot be reconstructed after delivery. |
| Traceability | Can an important result be connected to the evidence, input, assumption, or action behind it? | A wrong number, recommendation, or change cannot be traced to its source. |
| Modularity | Can one part be changed without opaque consequences elsewhere? | Small revisions trigger hidden dependencies or require broad regeneration. |
| Changeability | Can a future person or future AI modify the artifact safely? | The artifact is operationally frozen even though it still functions. |

## 5.1 Observability

Observability preserves externally visible evidence of execution. In software, this can include diffs, logs, test results, tool calls, and deployment records. In research or analysis, it can include source lists, retrieval records, transformations, and generated intermediate artifacts. In customer-service or operational agents, it can include state changes, messages sent, and policy-relevant actions. Agentic provenance systems such as PROV-AGENT reflect the same need to connect agent interactions to workflow context and downstream effects (Souza et al., 2025).

For agentic systems, this includes side effects in external state, not only the final response. An agent may update CRM tags, change access permissions, send external messages, or alter production configuration while still completing its nominal task successfully. If the triggering rule, policy state, and resulting state change are not preserved, the workflow can accumulate decision debt even when the task outcome appears correct.

Observability does not mean recording everything. Excessive traces can create their own storage, privacy, and review burdens. The goal is to preserve the externally relevant events needed to diagnose and intervene later.

## 5.2 Traceability

Traceability links a result to the evidence or action that supports it. The W3C provenance model treats derivation and the activities that produced an entity as core provenance relationships (W3C, 2013). In AI-generated work, traceability should answer questions such as: Which source supports this claim? Which input produced this number? Which policy authorized this action? Which code change introduced this behavior? Which assumption changed when the recommendation changed?

Traceability is particularly important when AI compresses many intermediate steps into a polished output. The more fluent the final artifact, the easier it can be to lose the distinction between supported conclusions and merely plausible prose.

### 5.3 Modularity

Modularity limits the blast radius of change. Sculley et al. (2015) describe how entanglement and boundary erosion can make ML systems difficult to modify safely. AI generation can reproduce the same problem at greater speed: adding code, dependencies, rules, or analytical transformations is cheap, while understanding how they interact remains expensive. A modular artifact exposes boundaries so that future modifications can be local rather than systemic.

For non-code artifacts, modularity can mean separating data, assumptions, calculations, narrative conclusions, and policy rules rather than blending them into a single opaque output. The objective is not aesthetic decomposition. It is to preserve the interfaces through which later change can occur.

### 5.4 Changeability and the future-changeability test

Changeability is the final property because it integrates the others into a practical test. An artifact may be observable but tightly coupled; traceable but impossible to modify without rerunning an entire workflow; modular but poorly documented. The relevant question is whether another competent actor can safely make a consequential change later.

Future-Changeability Test. Can a future person - or future AI - understand enough of the artifact's evidence, assumptions, dependencies, and interfaces to modify it safely without reconstructing the work from scratch?

The test is intentionally qualitative. This paper does not propose a numeric decision-debt score because the cost of future reconstruction is domain-specific and because fixed weights would imply a calibration that has not been empirically established.

**Table 3. Qualitative decision-debt checklist**

| Decision-debt level | Typical signals | Interpretation |
|---|---|---|
| Low | Key assumptions and sources are preserved; interfaces are explicit; consequential dependencies and state changes are reconstructable. | A future actor can understand and modify the artifact without substantial rediscovery. |
| Moderate | Some reconstruction is possible, but important assumptions, evidence links, or dependencies require manual rediscovery. | Future change remains feasible but carries meaningful delay and error risk. |
| High | The output works, but consequential assumptions, evidence, state changes, or interfaces cannot be reliably reconstructed. | Future actors may need to recreate the analysis or workflow before making a safe change. |

## 6. External Reconstruction, Not Private Reasoning

A tempting response to opaque AI-generated work is to demand the model's complete reasoning trace. That is the wrong control target. Private chain-of-thought can be unavailable, unstable, misleading, or inappropriate to expose. More importantly, future maintenance usually does not require every internal token that contributed to a result. It requires enough external evidence to reconstruct consequential structure.

External reconstruction evidence can include original sources, tool outputs, diffs, tests, schemas, assumptions, acceptance criteria, version identifiers, policy decisions, and interfaces. Provenance standards demonstrate that useful reconstruction can be built around entities, activities, agents, and derivation relationships rather than an internal narrative of cognition (W3C, 2013). Modern ML and agentic provenance work likewise focuses on captureable workflow events and artifacts (Schlegel & Sattler, 2025; Souza et al., 2025).

Principle 1 (External Reconstruction Principle). Reliable maintenance does not require access to private model reasoning; it requires sufficient external evidence to reconstruct the consequential structure of the work.

This principle also connects decision debt to verification independence. Independent verification is stronger when a reviewer can return to original evidence, tests, or ground truth rather than merely inspect the

producer's own summary. Preserving reconstruction evidence therefore serves both immediate verification and later changeability (Kao, 2026c).

## 7. Operationalizing the Framework

The framework is intentionally lightweight. It is not a proposal to preserve every prompt or turn every AI workflow into a full provenance database. The preservation depth should reflect the expected lifetime and consequence of the work. Table 3 illustrates how the four properties can be implemented across three common domains.

A compact analytical example illustrates the distinction. Consider an AI-generated investment memo that reports a plausible NPV and a clear recommendation. If the discount-rate assumption, data vintage, source-to-claim links, and sensitivity model are missing, the memo may be correct today yet expensive to revise when market conditions change. The missing structure creates decision debt even if the recommendation initially survives review.

A lower-debt version need not preserve every prompt. It preserves the versioned inputs, explicit assumptions, calculation artifact, primary-source links, and parameters that can be changed independently. The future analyst can then update one assumption and understand which conclusions should move with it rather than recreating the analysis from scratch.

**Table 4. Illustrative preservation patterns across AI-generated work**

| Domain | Preserve | Why it matters later |
|---|---|---|
| Software | Diffs, tests, dependency changes, interface contracts, relevant tool output, migration assumptions. | A future maintainer can locate what changed, test behavior, understand dependencies, and modify a bounded component. |
| Research / analysis | Primary sources, claim-source links, data transformations, key assumptions, calculation artifacts, versioned inputs. | A future analyst can update evidence or assumptions without recreating the entire analysis. |
| Agent workflows | Action logs, external state changes, policy/authority boundary used, acceptance evidence, exception/escalation records. | A future operator can reconstruct why the agent acted, what state changed, and which parts of the workflow can be safely rerun or amended. |

Three implementation implications follow. First, preservation should focus on consequential interfaces. If an AI-generated component can affect production state, money, external communications, or reusable analytical conclusions, its boundary and evidence deserve stronger preservation than ephemeral drafting steps. Second, artifacts should be versioned together with the evidence required to interpret them. A provenance record that points to mutable or unavailable sources may not support future reconstruction. Third, review should include a maintenance question before acceptance: what would a future actor need if one important assumption changed?

**Table 5. Risk- and lifetime-adjusted preservation depth**

| Work profile | Preservation depth | Illustrative practice |
|---|---|---|
| Ephemeral / low-stakes | Minimal | Retain final output and only the evidence needed for immediate review. |
| Reusable / medium-consequence | Structured | Preserve key assumptions, source links, versioned inputs, important actions, and interfaces. |
| Long-lived / high-consequence | Strong | Preserve end-to-end reconstruction evidence, consequential state changes, validation artifacts, dependencies, and change interfaces. |

The principle is not to minimize decision debt everywhere. Preservation depth should scale with expected artifact lifetime, consequence, and the probability of future change. Disposable drafting can rationally retain little; long-lived or high-consequence work should preserve much more.

These implications are consistent with recent quality-assurance research. Sun et al. (2026) show why passing tests is insufficient as a complete quality criterion for generated code. Schlegel and Sattler (2025) show why fragmented artifact stores weaken end-to-end provenance. Souza et al. (2025) show that agentic workflows require richer provenance relationships than conventional workflow logging. Decision debt connects these concerns to the practical requirement that the accepted artifact remain changeable.

## 8. Integration with Delegation, Oversight, and Verification

Decision debt is the fourth layer in a broader task-governance sequence developed across the author's prior working papers. The layers address different questions and should not be collapsed into one framework.

Delegation Contracts define what the agent is asked to accomplish, what counts as acceptable completion, what authority is granted, and when control returns to another actor (Kao, 2026a). Attention-Constrained Oversight determines where scarce human attention should be allocated based on consequence and autonomy boundaries (Kao, 2026b). Verification Independence addresses whether review has a sufficiently distinct basis for detecting the producer's errors (Kao, 2026c). Decision Debt addresses what must be preserved after acceptance so that the resulting work can still be inspected and changed later.

The sequence can be summarized as four questions: Was the work properly delegated? Was attention allocated to the right decisions? Was the result independently verified where necessary? Can the accepted artifact still be reconstructed and changed later? A workflow can succeed on the first three questions and still accumulate decision debt if it discards the evidence and structure required for future intervention.

## 9. Limitations and Research Agenda

The framework is conceptual and has several limitations. First, decision debt is not yet an empirically calibrated construct. The four preservation properties are intended as diagnostic dimensions, not validated scales. Future work could operationalize them through observable proxies such as missing source links, undocumented dependency changes, reconstruction time, change-failure rate, or the fraction of consequential decisions that can be traced to external evidence. Candidate empirical proxies include reconstruction time relative to recreating the work from scratch, the share of consequential claims linked to external sources, the proportion of external state changes with reconstructable triggers, and change-failure rates after a requirement update. These are research measures, not validated production KPIs.

Second, preservation creates costs. Logging, provenance capture, modularization, and documentation consume storage, engineering effort, latency, and human attention. The optimal preservation level will therefore depend on artifact lifetime, consequence, regulatory requirements, and expected frequency of change. A short-lived draft should not be governed like production infrastructure.

Third, preserving more information can create privacy and security risks. Provenance systems may expose sensitive data, credentials, user activity, or protected reasoning inputs. W3C provenance guidance itself notes that provenance access can reveal sensitive interests and information (W3C, 2013). Future research should therefore treat selective disclosure and retention policy as part of decision-debt management rather than assuming that maximum logging is always desirable.

Fourth, the framework generalizes beyond software, but current empirical evidence is strongest in software engineering and ML provenance. Studies are needed in research synthesis, financial analysis, operations, and other domains where AI-generated decisions are maintained over time. A useful experiment would compare two workflows that achieve similar initial task quality but preserve different levels of external reconstruction evidence, then measure the time, error rate, and confidence associated with a later requirement change.

Finally, decision debt can accumulate across versions. Even well-preserved artifacts may become difficult to change when successive AI-generated revisions overwrite assumptions, replace sources, or add dependencies faster than governance processes update them. Longitudinal studies should therefore examine not only single artifacts but debt accumulation and repayment across repeated AI-assisted changes.

## 10. Conclusion

AI-generated work can be correct and still be expensive to live with. The difference becomes visible only when the environment changes, an assumption fails, a source is updated, a dependency breaks, or a future actor needs to understand why the artifact looks the way it does.

Decision debt names the future cost created when accepted AI-generated work does not preserve the assumptions, evidence, dependencies, and interfaces needed for later reconstruction and modification. The proposed framework organizes this problem around four preservation properties: observability, traceability, modularity, and changeability. These properties connect established work on technical debt and provenance to a broader lifecycle question for AI-generated artifacts.

The practical implication is not to record everything and not to demand access to private model reasoning. It is to preserve enough external evidence and structure that future actors can intervene intelligently. An AI-generated artifact can be functionally correct today and still create decision debt if future actors cannot reconstruct what matters well enough to change it safely. Correctness is a present-tense property; changeability is a future-tense property. Reliable AI-generated work requires both.

## AI Use Disclosure

OpenAI's ChatGPT was used only for literature discovery and bibliographic verification. The decision-debt framework and underlying concepts originate in the author's prior published work (Kao, 2026d). The author independently reviewed all cited sources and remains solely responsible for the final manuscript, its arguments, source selection, and citations.

## References


Cotroneo, D., Improta, C., & Liguori, P. (2025). Human-written vs. AI-generated code: A large-scale study of defects, vulnerabilities, and complexity. 2025 IEEE 36th International Symposium on Software Reliability Engineering (ISSRE), 252-263. https://doi.org/10.1109/ISSRE66568.2025.00035

Kao, J. (2026a). Delegation contracts for agentic AI: A task-level framework for objectives, context, constraints, acceptance criteria, authority, and escalation. Working paper.

Kao, J. (2026b). Attention-constrained oversight for agentic AI: Risk, reversibility, authority, and human attention. Working paper.

Kao, J. (2026c). Verification independence for agentic AI: Designing review beyond self-correction and multi-agent agreement. Working paper.

Kao, J. (2026d). How to stay in control when AI does the work: A practical framework for delegation, judgment, and accountability. Independent publication.

Kruchten, P., Nord, R. L., & Ozkaya, I. (2012). Technical debt: From metaphor to theory and practice. IEEE Software, 29(6), 18-21. https://doi.org/10.1109/MS.2012.167

Molison, A. S., Moraes, M., Melo, G., Santos, F., & Assuncao, W. K. G. (2025). Is LLM-generated code more maintainable & reliable than human-written code? 2025 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), 151-162. https://doi.org/10.1109/ESEM64174.2025.00036

Nunes, H., Figueiredo, E., Rocha, L., Nadi, S., Ferreira, F., & Esteves, G. (2025). Evaluating the effectiveness of LLMs in fixing maintainability issues in real-world projects. 2025 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). https://doi.org/10.1109/SANER64311.2025.00069

Sawada, S., Shirai, T., Kashiwa, Y., Yamaguchi, K., Iwata, H., & Iida, H. (2026). To what extent does agent-generated code require maintenance? An empirical study. 30th International Conference on Evaluation and Assessment in Software Engineering (EASE 2026), Short Papers and Emerging Results. https://doi.org/10.48550/arXiv.2605.06464

Schlegel, M., & Sattler, K.-U. (2025). Capturing end-to-end provenance for machine learning pipelines. Information Systems, 132, 102495. https://doi.org/10.1016/j.is.2024.102495

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. Advances in Neural Information Processing Systems, 28, 2503-2511. https://papers.nips.cc/paper_files/paper/2015/hash/86df7dcfd896fcaf2674f757a2463eba-Abstract.html

Souza, R., Gueroudji, A., DeWitt, S., Rosendo, D., Ghosal, T., Ross, R., Balaprakash, P., & Ferreira da Silva, R. (2025). PROV-AGENT: Unified provenance for tracking AI agent interactions in agentic workflows. 2025 IEEE International Conference on eScience (eScience). https://doi.org/10.1109/eScience65000.2025.00093

Sun, X., Ståhl, D., Sandahl, K., & Kessler, C. (2026). Quality assurance of LLM-generated code: Addressing non-functional quality characteristics. Journal of Systems and Software, 238, 112885. https://doi.org/10.1016/j.jss.2026.112885

World Wide Web Consortium. (2013). PROV model primer. W3C Working Group Note. https://www.w3.org/TR/prov-primer/