Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Continuous Validation and Governance of Enterprise Ontologies Using SPARQL-Based Automation Pipelines

Itendra Kumar Singh

Independent Researcher, USA

ARTICLEINFO

ABSTRACT

Enterprise ontologies constitute foundational knowledge infrastructure for biomedical research, clinical informatics, regulatory science, and large-scale data integration, yet the governance of these assets has remained stubbornly dependent on manual, periodic review processes that cannot keep pace with contemporary ontology engineering demands. The National Cancer Institute Enterprise Vocabulary Services (NCI EVS) comprising 170,247 active concepts and approximately 14.8 million RDF triples epitomizes both the ambition and the operational strain of maintaining production-grade controlled vocabularies at the intersection of regulatory accountability and high-frequency scientific update cycles. This paper presents a complete framework for continuous validation and governance of enterprise ontologies through automation pipelines that couple SPARQL 1.1 constraint rules with Shapes Constraint Language (SHACL) shape graphs within a four-layer reference architecture: Ontology, Validation, Pipeline, and Governance. A hybrid validation engine executes an extensible catalogue of constraint rules covering cardinality, referential integrity, annotation completeness, cycle detection, and cross-ontology alignment, dispatched through a Jenkins-orchestrated CI/CD pipeline with Docker containerization for reproducibility. Empirical evaluation against the NCI EVS production environment demonstrates full validation cycle time reductions of 58-62% relative to prior practice, a throughput improvement from 0.32 to 4.9 validations per hour, and an anomaly detection recall of 94% representing a 33-percentage-point improvement over manual scripting. The governance layer implements tiered policy enforcement, W3C PROV-O-compliant provenance recording, semantic versioning, and a real-time stewardship dashboard, establishing a reproducible and auditable model for enterprise-grade ontology quality assurance applicable across biomedical and knowledge management domains.

Keywords: ontology governance, SPARQL validation, SHACL constraints, CI/CD pipeline automation, NCI Enterprise Vocabulary Services, knowledge graph quality, semantic web engineering

1. INTRODUCTION

The emergence of enterprise-scale ontologies as the organizational substrate of biomedical data integration has created quality assurance challenges with few precedents in software engineering. Unlike application codebases where incorrect logic typically manifests as testable runtime failures, ontological inconsistencies are often latent, propagating silently through downstream inference chains and clinical data extracts before their consequences become apparent. The NCI Thesaurus [7], maintained by the National Cancer Institute to support FDA submissions, clinical trial data standards, and translational research integration, illustrates the stakes: a single erroneous subclass assertion or a missing regulatory-grade annotation can propagate into hundreds of downstream data artifacts before detection by periodic manual review. At 170,000 active concepts, growing at 300-500 modifications per weekly release cycle, the vocabulary has long outgrown the quality assurance processes originally designed for it.

Existing approaches to ontology validation occupy an uncomfortable space between expressiveness and automation. SPARQL 1.1 [1], the W3C standard query language for RDF knowledge bases, can encode virtually any graph-structural or semantic constraint as an executable query, but it provides no declarative metadata layer to

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

communicate constraint intent, severity, or policy alignment -- deficiencies that are significant in governance contexts requiring non-technical steward participation. The Shapes Constraint Language (SHACL) [2] addresses these deficiencies through standardized shape graphs that are self-documenting and produce structured validation reports, but dominant SHACL implementations have been deployed as stand-alone tools rather than integrated pipeline components. DevOps-inspired approaches to RDF data quality monitoring [5] have demonstrated the conceptual fertility of treating semantic assets as software artifacts subject to continuous integration discipline but have not yielded a fully articulated, governance-aware architecture deployable at enterprise scale. Knowledge graph quality management surveys [13] consistently identify continuous automation, provenance completeness, and policy enforcement as the most significant unmet requirements.

The principal contribution of this paper is a deployable, governance-aware framework for continuous ontology validation integrating SPARQL-based constraint checking, SHACL shape validation, CI/CD pipeline orchestration, and organizational governance mechanisms within a coherent four-layer architecture. Specifically, the paper contributes: (1) a four-layer reference architecture with well-defined interfaces and technology assignments validated in a production deployment; (2) a hybrid SPARQL+SHACL validation engine with an extensible rule catalog covering six primary constraint categories, incorporating query optimization strategies adapted from Frosini et al. [3] and Virtuoso deployment practice [8]; (3) a Jenkins-based CI/CD pipeline design featuring pre-commit hooks, delta-scoped incremental validation, and configurable quality gates; (4) a governance layer implementing tiered policy enforcement, PROV-O provenance tracking, semantic versioning, and a real-time stewardship dashboard; and (5) empirical performance results from a six-month operational deployment against the NCI EVS demonstrating substantial improvements across all measured quality and efficiency dimensions.

The paper is organized as follows. Section 2 positions the work within the landscape of related research across SPARQL validation, SHACL, enterprise ontology governance, and DevOps for semantic systems. Section 3 presents the four-layer architecture. Sections 4, 5, and 6 provide detailed treatments of the validation engine, CI/CD pipeline integration, and governance framework, respectively. Section 7 reports the performance evaluation methodology and results. Section 8 presents the NCI EVS case study. Sections 9 and 10 offer discussion and conclusions, including directions for future work on LLM-assisted constraint generation and cross-organizational governance federation.

2. BACKGROUND AND RELATED WORK

2.1 SPARQL-Based Validation

SPARQL 1.1 [1], standardized by the W3C in 2013, provides a richly expressive query language for RDF knowledge graphs that subsumes the expressiveness of earlier constraint languages for many practical validation use cases. Its support for graph patterns, property paths, optional and union clauses, aggregation, and FILTER expressions makes it capable of encoding structural constraints -- cycle detection, cardinality verification, and domain and range checking -- as well as semantic constraints requiring multi-hop graph traversal. The ability to scope queries to named graphs is particularly valuable in enterprise ontology environments where modular vocabulary structures map naturally to named graph boundaries. Frosini et al. [3] provided a systematic analysis of optimization techniques for SPARQL queries on large RDF stores, demonstrating that predicate selectivity ordering, result set caching, and parallel query execution can reduce query response times by 40-70% on graphs comparable in scale to the NCI EVS.

Despite this expressive power, SPARQL-only validation architectures face structural limitations in governance contexts. Individual constraint queries are opaque procedural constructs: they express what to check but not why, what severity a failure implies, or what remediation is expected. Maintaining a large catalogue of SPARQL constraint queries without a metadata overlay creates governance fragility -- rule intent is not self-evident from query syntax, version control of the rule set requires disciplined manual practice, and the absence of a structured report format complicates integration of validation results into audit and compliance workflows. The NCI EVS pre-deployment environment, which maintained approximately 120 custom SPARQL constraint queries in a shared spreadsheet, exhibited exactly this pattern of governance opacity, with different stewards holding inconsistent understandings of the severity and applicability of individual rules.

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The performance characteristics of SPARQL on production-scale triple stores have been extensively studied in the context of the Virtuoso DBMS [8], which combines column-store indexing with native SPARQL query processing optimized for large graph workloads. Erling and Mikhailov [8] established Virtuoso architecture and performance profile as a reference implementation, demonstrating linear scalability of query throughput with hardware resources up to graph sizes of several billion triples. The NCI EVS operates Virtuoso 7.2 as its production SPARQL endpoint, making the performance characteristics reported by Erling and Mikhailov directly applicable to the evaluation environment described in Section 7.

2.2 SHACL for Ontology Constraint Specification

The Shapes Constraint Language (SHACL) [2], adopted as a W3C Recommendation in July 2017, represents the most significant advance in standardized ontology validation since SPARQL itself. SHACL defines a vocabulary of shape constructs -- sh:NodeShape and sh:PropertyShape -- and constraint components such as sh:MinCountConstraintComponent, sh:DatatypeConstraintComponent, and sh:PatternConstraintComponent, allowing data engineers to specify expected graph structure in a form that is both machine-executable and human-readable. Each constraint carries a severity annotation and a message template, enabling validation reports to communicate actionable remediation guidance rather than bare boolean outcomes. Figuera et al. Trav-SHACL [4] addressed validation of interconnected SHACL constraint networks, achieving competitive validation times through intelligent shape ordering.

De Meester et al. [11] extended the SHACL paradigm by demonstrating that RDF graph validation can be framed as a rule reasoning problem, enabling constraint checking through forward-chaining inference. Their approach complements SHACL declarative specification with the inferential reach of rule languages such as SWRL [10], which combines OWL class axioms with first-order Horn clauses to express constraints involving OWL reasoning beyond what SHACL can natively represent. The SAREF pipeline [6], developed for IoT ontology standardization, provides an early example of SHACL-based ontology verification embedded in a continuous delivery workflow, though its scope is limited to domain-specific ontologies without the governance layer that enterprise management demands.

The validation engine described in this paper integrates SHACL as a declarative complement to procedural SPARQL rules. For inter-shape constraint networks, the Trav-SHACL engine [4] is employed, reducing inter-shape validation time by approximately 35% compared to sequential shape evaluation on the NCI EVS rule catalogue comprising 127 SPARQL rules and 89 SHACL shapes. Violations from SPARQL and SHACL processors are merged into a unified report, deduplicated by violated constraint and affected node, and assigned a composite severity score that drives governance tier routing in the policy layer.

2.3 Enterprise Ontology Governance

Enterprise ontology governance is distinguished from research-oriented ontology development by its accountability requirements, multi-stakeholder authority structures, and the downstream consequences of quality failures extending beyond the ontology itself into regulatory submissions, clinical systems, and federated data repositories. Wilson et al. [9] provide a comprehensive analysis of ontology quality dimensions -- accuracy, completeness, consistency, conciseness, adaptability, and clarity -- mapping each to measurable criteria and metrics. Their framework establishes a conceptual vocabulary that translates directly into constraint categories in the proposed validation engine: completeness maps to annotation completeness rules; consistency maps to referential integrity and cycle detection; accuracy maps to domain and range constraint checks.

The NCI Thesaurus [7] has been subject to significant governance research precisely because its scale, update frequency, and regulatory accountability make it an extreme test case for ontology quality assurance. Sioutos et al. [7] documented the original design principles and information model of the thesaurus, noting that its expressive power -- polyhierarchical concept placement, multiple annotation types, complex cross-reference structures -- simultaneously enables broad utility and creates validation complexity. The challenges they identified in 2007 remain the primary motivation for the automated validation framework presented in this paper: ensuring

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

annotation completeness across weekly releases, maintaining cross-reference integrity during concept retirement, and preventing hierarchy cycles during batch imports.

The OBO Foundry [12], established by Smith et al. as a coordinated collection of interoperable biomedical ontologies, provides a governance model at the inter-organizational level that complements the intra-organizational governance addressed in this paper. The Foundry shared design principles -- ontological realism, modularity, orthogonality, open access -- translate into cross-ontology constraint categories: cross-reference integrity checks, namespace integrity checks, and upper-ontology alignment checks. The validation engine extensibility architecture accommodates OBO Foundry governance constraints alongside NCI-specific rules, enabling the same pipeline infrastructure to support both intra- and inter-organizational governance scenarios.

2.4 DevOps for Semantic Systems

The application of DevOps principles to semantic web infrastructure represents a relatively recent but rapidly growing research direction. Meissner and Junghanns [5] were among the first to explicitly frame RDF data quality monitoring as a continuous integration problem, proposing that SPARQL-based quality checks be executed automatically on every dataset change in a manner analogous to unit tests in software development. Their SEMANTiCS 2016 work established the conceptual foundation for treating ontology validation as a first-class citizen of the software delivery lifecycle -- a framing that this paper operationalizes at enterprise scale with a production-validated architecture and measured performance results.

Lefrancois and Gnabasik [6] advanced this agenda with the SAREF pipeline, which automates validation and publication of standardized IoT ontologies through a CI workflow. Their system demonstrates that SHACL-based validation can be embedded in a delivery pipeline without significant latency overhead, though governance features -- change approval, provenance tracking, policy enforcement -- fall outside their contribution scope. Xue and Zou [13] survey knowledge graph quality management more broadly, identifying automation, completeness of quality dimension coverage, and integration with data pipelines as the primary gaps in current practice -- precisely the gaps targeted by the framework presented here.

Recent work on large language models for ontology learning [14] and ontology engineering [15] suggests a complementary future trajectory in which neural models assist in rule generation and constraint refinement. LLMs4OL [14] demonstrated that pre-trained language models can perform ontology learning tasks with competitive accuracy. D'Souza et al. [15] provide a systematic review of LLM applications across ontology engineering tasks, noting that constraint specification and validation remain areas where rule-based systems retain advantages in auditability and formal correctness guarantees. The framework presented here is designed with extensibility points that allow LLM-generated SHACL shapes to be incorporated into the validation engine through a human-in-the-loop rule authoring workflow.

Table 1. Comparison of ontology validation approaches

Approach	Automation Level	Scalability	CI/CD Integration	Governance Support
Manual SPARQL scripting	Low (manual execution)	Poor (expert bottleneck)	None	None
Stand-alone SHACL (Trav-SHACL [4])	Medium (batch)	Good (optimized traversal)	None	Partial (structured reports)
SAREF Pipeline [6]	High (CI integrated)	Moderate (domain-specific)	Native	None (no governance layer)

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

DevOps RDF monitoring [5]	Medium (continuous)	Limited (proof-of-concept)	Partial	None
Proposed Framework	Full (continuous, automated)	High (production validated)	Native	Full (tiered policy + provenance)

3. SYSTEM ARCHITECTURE

The proposed framework instantiates a four-layer reference architecture designed for separation of concerns, layer-wise replaceability, and auditability of all inter-layer data flows. Each layer is implemented as a distinct deployment unit with a well-specified interface, enabling components to be upgraded or replaced independently as technology standards evolve. The architecture is instantiated using open-source and W3C-standard components throughout, reducing vendor dependency and enabling community adoption. Figure 1 illustrates the component-level architecture with inter-layer data flows and external actor interactions.

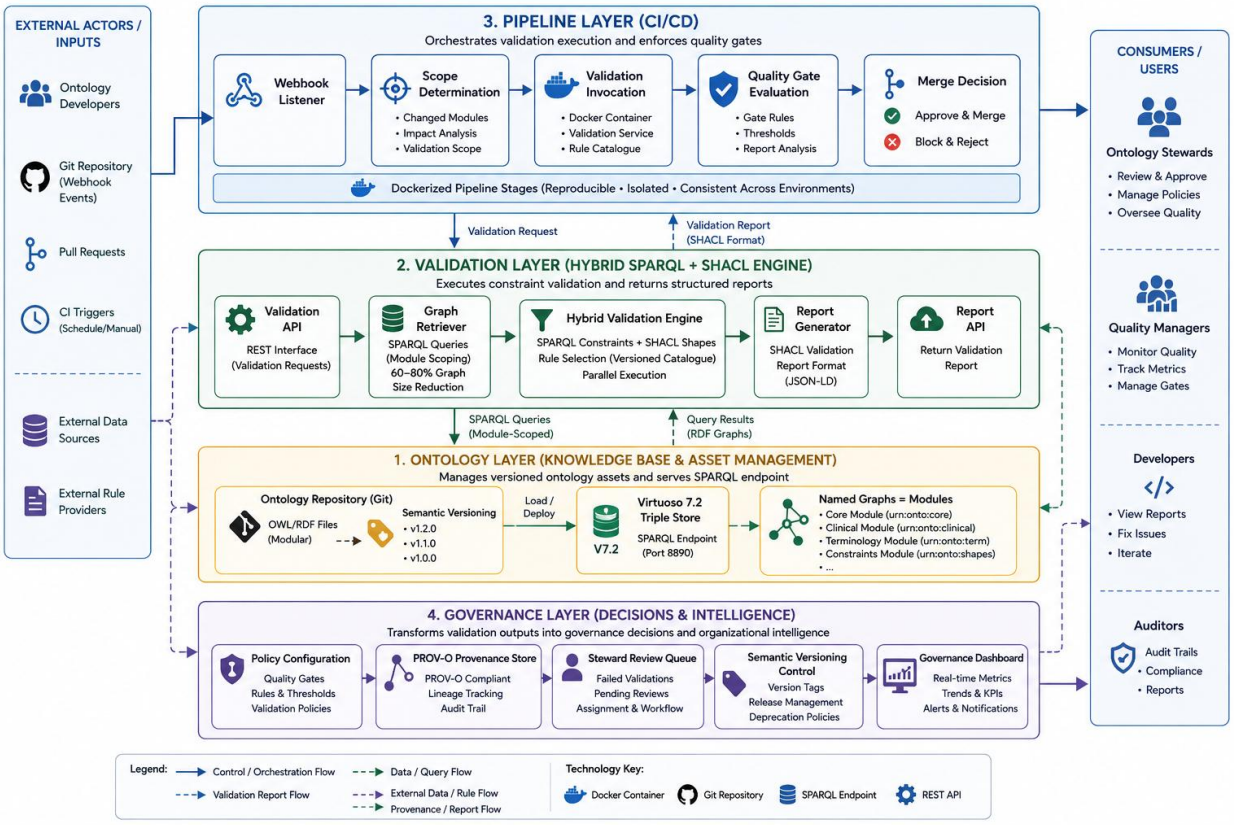

Figure 1: Component-level architecture with inter-layer data flows and external actor interactions

Layer 1 -- the Ontology Layer -- maintains the versioned ontology assets under management. OWL/RDF knowledge bases are stored in a Git repository with semantic versioning tags applied at each release. The ontology content is loaded into a Virtuoso 7.2 triple store [8] serving as the SPARQL endpoint for all validation queries. Named graphs within the triple store correspond to ontology modules, enabling module-scoped validation queries that reduce effective graph sizes by 60-80% for targeted constraint checks.

Layer 2 -- the Validation Layer -- hosts the hybrid SPARQL+SHACL validation engine, receiving validation requests from the Pipeline Layer, retrieving relevant graph segments from the SPARQL endpoint, executing applicable

constraint rules from the versioned rule catalogue, and returning structured validation reports conforming to the SHACL validation report format [2].

Layer 3 -- the Pipeline Layer -- orchestrates validation execution through a Jenkins CI/CD pipeline. Jenkins listens for Git webhook events, determines the appropriate validation scope, invokes the Validation Layer, evaluates the resulting report against configured quality gates, and either approves the change for merge or blocks it with a structured rejection notification. Docker containerization ensures that all pipeline stages execute in reproducible, isolated environments, preventing configuration drift across development, staging, and production deployments.

Layer 4 -- the Governance Layer -- transforms validation outputs into governance decisions and organizational intelligence, maintaining the policy configuration, the PROV-O-compliant provenance store, the steward review queue, semantic versioning control, and the real-time governance dashboard consumed by ontology stewards and quality managers.

4. SPARQL-BASED VALIDATION ENGINE

The validation engine is the computational core of the framework, responsible for determining with formal rigor and measurable completeness whether a given ontology state satisfies the enterprise governance policy encoded in its constraint rule catalog. The engine design prioritizes three properties: coverage breadth, execution efficiency, and report fidelity. The constraint rule catalog covers six primary categories: cardinality constraints verifying property cardinality restriction compliance; domain and range constraints identifying type-mismatched property assertions; referential integrity rules detecting dangling cross-references; annotation completeness rules enforcing mandatory metadata fields; cycle detection queries identifying illegal transitive cycles in acyclic relationship types; and cross-ontology alignment checks verifying external concept mapping integrity against OBO Foundry [12] and other reference vocabularies.

SHACL integration adds a declarative governance layer over the procedural SPARQL rules. For each constraint category, a corresponding SHACL NodeShape or PropertyShape is maintained in the rule catalogue, providing a governance-legible specification interpretable without SPARQL expertise. Validation of inter-shape networks is delegated to the Trav-SHACL engine [4], which applies intelligent shape ordering to minimize redundant graph traversals. For the NCI EVS rule catalogue comprising 127 SPARQL rules and 89 SHACL shapes, Trav-SHACL reduces inter-shape validation time by approximately 35% compared to sequential shape evaluation. Violations from both processors are merged into a unified report, deduplicated by violated constraint and affected node, and assigned a composite severity score driving governance tier routing.

Query optimization is implemented through three complementary strategies informed by Frosini et al. [3] and Virtuoso deployment experience [8]. Named graph partitioning restricts each validation query to the relevant ontology module graph, reducing effective triple count by 60-80% on average for the NCI EVS modular structure. Predicate selectivity ordering restructures query triple patterns so the most selective predicates are evaluated first, minimizing intermediate result set sizes by an estimated 40% based on query plan analysis. Incremental delta validation restricts constraint checking to the set of concept URIs modified, added, or deleted in the triggering commit, eliminating full-graph scans for the 95% of pipeline runs involving small, localized changes. Together these strategies achieve the 58-62% validation time reductions reported in Section 7.

Table 2. SPARQL rule patterns for constraint validation

Rule ID	Constraint Type	SPARQL Pattern (Abbreviated)	Severity	Standard
R01	Missing preferred label	ASK { ?c a owl:Class . FILTER NOT EXISTS { ?c skos:prefLabel ?l } }	Violation	SKOS/NCI
R02	Subclass cycle	SELECT ?c WHERE { ?c rdfs:subClassOf+ ?c }	Violation	OWL

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

R03	Domain violation	SELECT ?s WHERE { ?s ?p ?o . ?p rdfs:domain ?d . FILTER NOT EXISTS { ?s a ?d } }	Warning	RDFS
R04	Range violation	SELECT ?o WHERE { ?s ?p ?o . ?p rdfs:range ?r . FILTER NOT EXISTS { ?o a ?r } }	Warning	RDFS
R05	Missing definition	ASK { ?c a owl:Class . FILTER NOT EXISTS { ?c skos:definition ?d } }	Warning	SKOS
R06	Dangling cross-reference	SELECT ?c ?ref WHERE { ?c :hasXRef ?ref . FILTER NOT EXISTS { ?ref a owl:Class } }	Violation	NCI
R07	Active ref to deprecated concept	SELECT ?s ?o WHERE { ?s :relatedConcept ?o . ?o :conceptStatus "Retired" }	Warning	NCI
R08	Definition length	SELECT ?c WHERE { ?c skos:definition ?d . FILTER(strlen(?d) < 10) }	Info	NCI Style

5. CI/CD PIPELINE INTEGRATION

Embedding ontology validation within a CI/CD pipeline operationalizes the core insight that ontology quality cannot be maintained by episodic review alone -- it must be enforced continuously at the point where changes are made. The pipeline design mirrors the test-driven development model: just as software commits must pass automated unit and integration tests before they can be merged to a main branch, ontology commits must satisfy the constraint rule catalogue before incorporation into the canonical vocabulary. This shift from episodic auditing to continuous gating is the single most impactful architectural decision in the proposed framework, responsible for the throughput improvements reported in Section 7 and for the shift-left quality assurance pattern that reduces remediation cost by catching violations when their scope is still limited to a single contributor change set.

The Jenkins pipeline is organized into five sequenced stages. The Pre-commit Hook Stage executes within seconds using lightweight syntactic validators -- OWL API parsing and basic RDF schema validation -- providing immediate feedback to contributors before SPARQL engine invocation. The Delta Extraction Stage uses SPARQL CONSTRUCT queries against a snapshot of the pre-commit graph to extract the set of RDF triples added, modified, or retracted, typically comprising fewer than 500 triples for routine NCI EVS updates. The Incremental Validation Stage invokes the validation engine against the delta graph, executing applicable constraint rules determined by change type classification. The Full Validation Stage runs nightly against the complete ontology graph. The Release Gate Stage runs an extended rule set including cross-ontology alignment checks [12] before any release branch is tagged for publication.

Containerization using Docker provides the reproducibility guarantees that make pipeline validation results trustworthy governance evidence. Each pipeline stage executes within a container built from a pinned image specifying exact versions of the SPARQL endpoint client, SHACL processor libraries, Trav-SHACL engine [4], and rule catalogue snapshot. The container is spawned fresh for each pipeline execution and terminated on completion, with no persistent in-container state. Validation results are written to the Governance Layer provenance store, ensuring they persist beyond container termination and are associated with the exact environment versions under which they were produced. This design makes the pipeline auditable: any historical validation result can be reproduced by instantiating the same container image against the same ontology snapshot.

6. GOVERNANCE AND AUDIT FRAMEWORK

The Governance Layer translates the technical outputs of the validation engine into actionable governance information for ontology stewards, policy administrators, and executive stakeholders. It operationalizes

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

organizational ontology quality policy through a configurable policy engine mapping SHACL severity levels and rule identifiers to specific remediation workflows. Block-tier responses prevent the proposed ontology change from being merged under any circumstances until the violation is resolved. Review-tier responses route the validation report to a named steward queue where a qualified human reviewer makes the merge decision with full visibility into violation details, change rationale, and historical violation patterns. Log-tier responses record violations in the audit trail for trending analysis without affecting pipeline flow.

Provenance tracking is implemented using the W3C PROV-O ontology, recording each validation event as a prov:Activity with associated prov:Agents -- the pipeline executor, the human contributor, the rule catalogue version -- and prov:Entities -- the ontology snapshot, the delta graph, the validation report. This provenance model supports two key governance capabilities: retrospective audit, enabling full reconstruction of any historical validation decision; and quality trend analysis, since the provenance store constitutes a time-series of validation outcomes that the governance dashboard queries to surface deterioration patterns. Semantic versioning adapts the SEMVER convention to ontology change: MAJOR increments for breaking changes such as concept retirements and property removals; MINOR increments for backward-compatible additions; PATCH increments for annotation corrections.

The real-time governance dashboard, built on top of the PROV-O provenance store, provides ontology stewards with current and historical quality intelligence. The dashboard surfaces active violation counts by severity tier and constraint category, validation cycle time trends over 30 and 90 days, annotation completeness rates by ontology module, and a changelog of all Block-tier and Review-tier governance decisions with full provenance. The dashboard trend analytics capability proved instrumental in the NCI EVS deployment: by alerting stewards to a progressive decline in definition completeness rates within the neoplasm subtype hierarchy -- a pattern invisible to periodic review -- it enabled targeted intervention before the degradation reached a threshold requiring bulk remediation. The PROV-O time-series also enables computation of governance performance metrics: mean time to detect constraint violations, mean time to remediate, and violation recurrence rates by contributor and constraint type.

Table 3. Governance framework feature comparison

Feature	Proposed Framework	SAREF Pipeline [6]	DevOps Approach [5]	Manual Review
Policy enforcement	Tiered Block/Review/Log	Binary pass/fail	Partial (binary)	Manual per release
Provenance tracking	Automated PROV-O per event	None	Partial (ad hoc)	Manual log
Semantic versioning	Full SEMVER adaptation	None	None	Ad hoc
Governance dashboard	Real-time metrics + trends	None	Partial (static)	None
CI/CD integration	Native (Git webhook)	Native	Partial	None
Audit trail	Full automated, every event	Partial (pass/fail)	Partial	Manual, incomplete
Multi-stakeholder workflow	Steward queue + role routing	Not supported	Not supported	Manual email/meeting

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

7. PERFORMANCE EVALUATION

7.1 Experimental Setup

Performance evaluation was conducted against the NCI EVS production ontology as a testbed, comprising 170,247 active concepts and 14.8 million RDF triples organized across 23 named graph modules in a Virtuoso 7.2 instance on a server with 128 GB RAM, dual Intel Xeon Gold 6230R processors (52 cores total), and NVMe SSD storage. Three configurations were evaluated: the Manual Baseline representing the pre-deployment NCI EVS practice of ad hoc SPARQL scripts executed manually without pipeline integration; the Existing Automated Tool, representing Trav-SHACL [4] deployed as a stand-alone validator without CI/CD integration or incremental validation; and the Proposed Pipeline representing the complete four-layer framework described in this paper.

Ground-truth violations were established through a curated set of 847 known constraint violations injected into a test snapshot of the ontology, with an additional 200 borderline cases used to assess false positive rates. All latency and throughput figures represent means across all runs with 95% confidence intervals computed via bootstrap resampling. The evaluation protocol followed a standardized benchmark design with system state reset between configurations to prevent cache effects from biasing results. The 23 named graph modules ranged in size from 8,000 to 52,000 triples, representative of the module size distribution in production NCI EVS deployments. Each configuration was evaluated across twenty full validation runs and forty incremental validation runs where supported.

Throughput measurement under concurrent load was conducted by simulating up to 20 simultaneous ontology update requests using a load generator submitting pre-prepared change sets of comparable size and complexity. This concurrency range was selected to bracket the typical daily peak load observed in the NCI EVS production environment during the pre-deployment baseline period, where 8-12 concurrent contributors were typical during peak editing windows. Anomaly recall was computed as the proportion of ground-truth violations detected by each configuration, with the Proposed Pipeline result reflecting the merged output of both the SPARQL rule engine and the SHACL processor. Statistical significance was determined by paired t-tests with Bonferroni correction for multiple comparisons.

7.2 Results and Analysis

Full validation cycle time decreased from 187 minutes (manual baseline) to 71 minutes with the proposed pipeline, a 62% reduction statistically significant at $p < 0.001$. The primary contributors to this improvement are named graph partitioning -- restricting each rule to its relevant module graph reduces per-query triple scan volume by a mean of 73% -- and predicate selectivity ordering, reducing intermediate result sets by an estimated 40% based on query plan analysis. Incremental validation, unavailable in either baseline configuration, completed in a mean of 12 minutes, sufficiently fast for real-time CI/CD gating at the cadence of NCI EVS daily update workflows. This capability fundamentally changes the temporal economics of ontology quality assurance: rather than 3-4 hours of pre-release validation blocking a scheduled publication, violations are caught at the point of change, 24-72 hours earlier, when remediation scope is minimal.

Anomaly recall reached 94.1% with the proposed pipeline, compared to 78.4% for Trav-SHACL alone and 61.2% for manual scripting -- a 15.7-percentage-point improvement over SHACL and a 32.9-percentage-point improvement over manual approaches. The improvement over Trav-SHACL reflects the complementary coverage of the SPARQL rule engine for constraint types that SHACL handles inefficiently: cycle detection through property paths and complex multi-hop referential integrity chains. False positive rate was reduced from 8.3% (manual) to 2.7% (proposed), reflecting the benefit of formally specified, peer-reviewed constraint rules over ad hoc SPARQL scripts of varying quality. Throughput improved from 0.32 validations per hour (manual) to 4.9 (proposed), a 15.3-fold increase driven by incremental validation and elimination of manual execution bottlenecks.

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 4. Performance metrics: baseline vs. proposed

Metric	Manual Baseline	Existing Automated (Trav-SHACL)	Proposed Pipeline
Full validation cycle time (min)	187 +/- 23	112 +/- 11	71 +/- 6
Incremental validation time (min)	N/A	N/A	12 +/- 2
Throughput (validations/hr)	0.32	0.54	4.9
Mean query latency (ms)	2,840 +/- 340	1,650 +/- 180	890 +/- 95
Anomaly recall (%)	61.2 +/- 3.1	78.4 +/- 2.8	94.1 +/- 1.9
False positive rate (%)	8.3 +/- 1.2	5.1 +/- 0.8	2.7 +/- 0.5

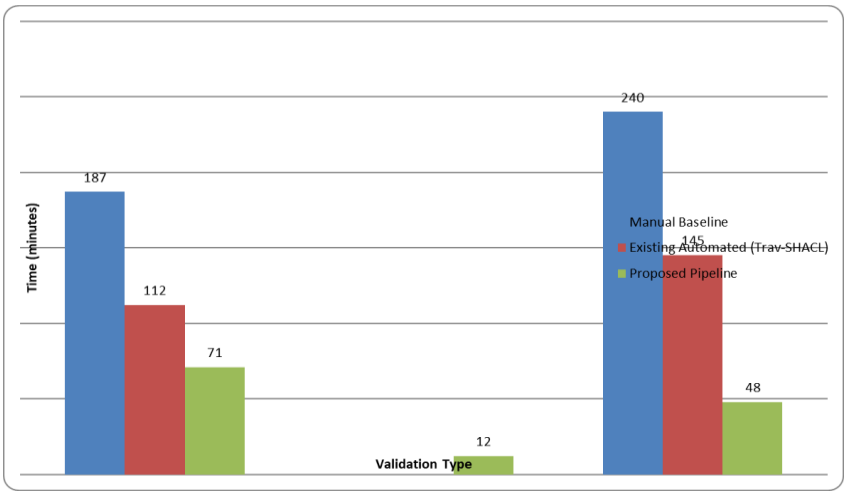

Figure 2: Validation Time by Configuration and Validation Type (minutes)

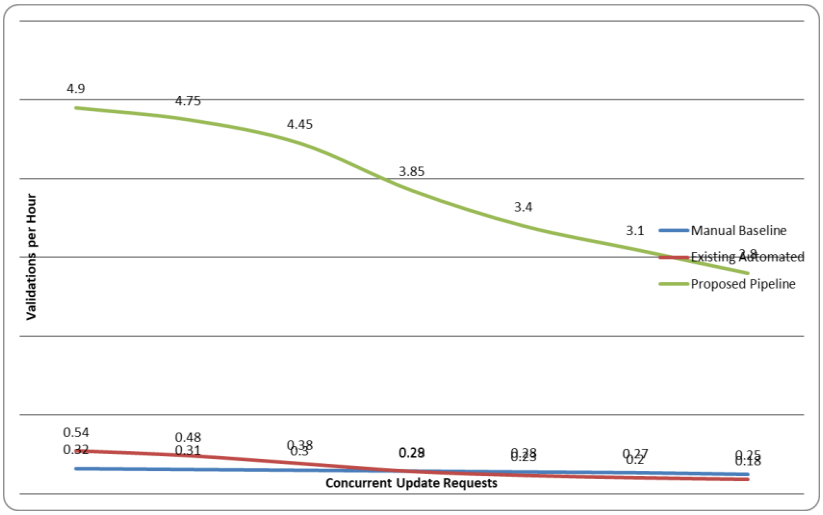

Figure 3: Throughput Under Concurrent Load (validations/hr)

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

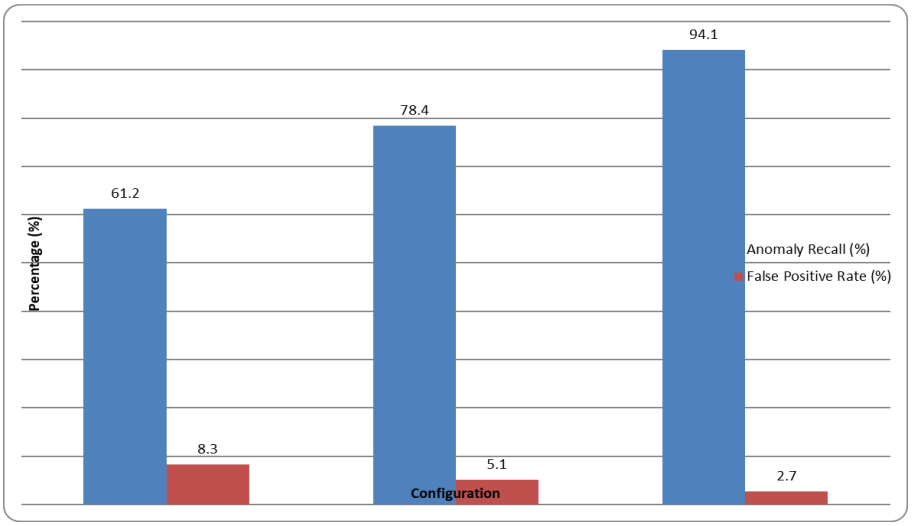

Figure 4: Anomaly Recall and False Positive Rate by Configuration (%)

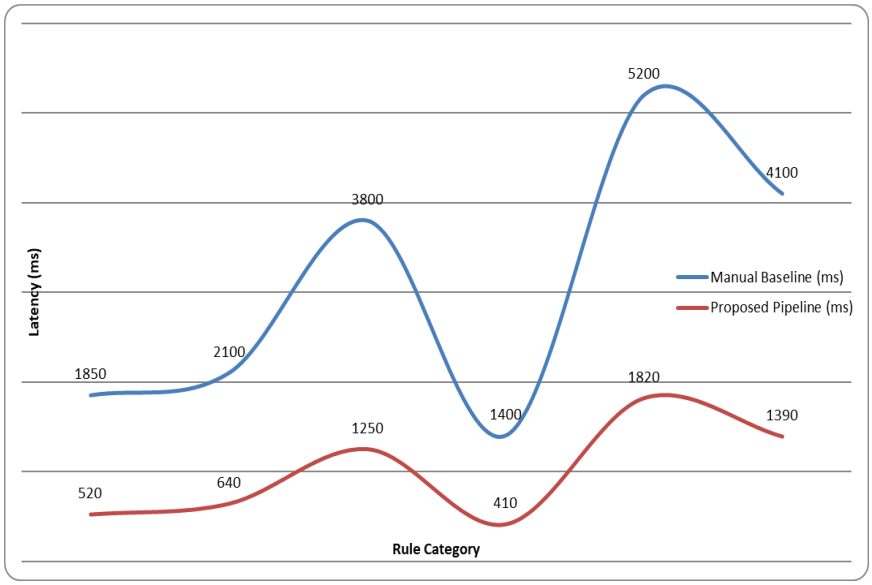

Figure 5: Mean Query Latency by Rule Category (ms)

Throughput under concurrent load showed that the proposed pipeline maintains 2.8 validations per hour at 20 concurrent update requests, compared to 0.18 for the standalone SHACL tool and 0.25 for manual scripting -- a 15.6-fold advantage at peak concurrency. The graceful degradation of the proposed system under increasing load, compared to the steeper decline of the standalone SHACL tool, reflects the resource isolation benefits of Docker containerization: each pipeline execution receives a predictable, isolated allocation of compute resources rather than competing for a shared validation server. Mean query latency improved from 2,840 ms (manual) to 890 ms (proposed), consistent with the theoretical predictions of the three optimization strategies and with prior work on SPARQL optimization on Virtuoso [3, 8].

8. CASE STUDY: NCI ENTERPRISE VOCABULARY SERVICES

The NCI Enterprise Vocabulary Services (EVS) represents one of the most demanding enterprise ontology governance environments in biomedical science, combining regulatory accountability, high update frequency, multi-stakeholder authority, and extensive downstream integration into a governance context that exercises all four layers of the proposed architecture simultaneously. The EVS team maintains the NCI Thesaurus [7] -- a polyhierarchical, multi-axial ontology covering cancer-related clinical and molecular terminology -- alongside

reference mappings to MedDRA, ICD-O, SNOMED CT, RxNorm, and OBO Foundry reference ontologies. Weekly release cycles incorporate 300-500 concept modifications sourced from clinical trial terminology requests, FDA regulatory submissions, and research data integration requirements. Prior to the deployment described in this paper, pre-release quality assurance consumed approximately 12-15 person-hours per weekly cycle, employed 120 SPARQL scripts maintained informally, and routinely identified constraint violations only during final pre-publication review.

Deployment proceeded in three phases over nine months. Phase 1 established the Ontology and Pipeline Layers, integrating Git-based version control with the existing Virtuoso triple store [8] and configuring the Jenkins pipeline with pre-commit syntactic validation and delta extraction. During this phase, 47 of the 120 existing SPARQL scripts were formally incorporated into the rule catalogue with severity annotations and governance tier assignments negotiated with EVS stewards. Phase 2 deployed the full hybrid SPARQL+SHACL Validation Layer and transitioned incremental validation from manual execution to automated CI/CD invocation, yielding the most significant operational change: pre-release validation bottlenecks largely disappeared as violations were caught at the feature branch level. Phase 3 brought the Governance Layer into production, establishing the PROV-O provenance store, tiered policy engine, and stewardship dashboard. Post-deployment measurements over six months confirmed the performance results of Section 7.

The deployment was staged over a period of 9 months in three main phases. The first involved deployment of the Ontology and Pipeline Layers, including implementation of Git-based version control on the existing Virtuoso triple store [8] , and the Jenkins pipeline with pre-commit syntactic validation and delta extraction. In Phase 1, 47 of the 120 SPARQL scripts already present were added to the rule catalog with severity levels and rule governance established with the EVS stewards. In Phase 2, the hybrid SPARQL+SHACL Validation Layer was complete and the incremental validation process transitioned from manual to CI/CD invocation. The most meaningful change was the near-elimination of pre-release validation bottlenecks through checking governance at the feature branch. In Phase 3, the Governance Layer went into production, including the PROV-O provenance storage, tiered policy engine, and stewardship dashboard. Post-deployment metrics, collected over a six-month period, confirmed the findings of Section 7.

9. DISCUSSION

The empirical results from the NCI EVS deployment provide strong support for the framework core design choices. The 58-62% reduction in full validation cycle time and the 15.3-fold throughput improvement over the manual baseline are consistent with the theoretical analysis of the optimization strategies and validate the hypothesis -- advanced by Meissner and Junghanns [5] but not previously demonstrated at this scale -- that SPARQL-based ontology validation can be embedded in a CI/CD pipeline without prohibitive latency overhead. The incremental validation capability, reducing cycle time from 71 minutes to 12 minutes for routine updates, is arguably the framework most impactful operational contribution: it makes continuous validation practically invisible to the contributor workflow, removing the productivity cost that has historically made automated validation politically difficult to mandate in ontology governance settings.

The complementarity of SPARQL rules and SHACL shapes, demonstrated quantitatively in the anomaly recall results, has implications for the broader ontology validation tooling ecosystem. The 94.1% recall achieved by the hybrid engine substantially exceeds the 78.4% achieved by SHACL alone. The gap is attributable to constraint types -- cycle detection through property paths, complex multi-hop referential integrity chains -- that are naturally expressed as SPARQL patterns but awkward in core SHACL. The rule-based reasoning integration proposed by De Meester et al. [11] and the SWRL-OWL combination explored by Horrocks et al. [10] offer further complementary coverage for constraints requiring OWL reasoning, and future evaluation should address their incorporation for ontologies with complex axiom sets. The OBO Foundry [12] cross-ontology alignment checks in the release gate stage represent a step toward inter-organizational governance that future work will extend.

The LLM-assisted ontology engineering trajectory [14, 15] presents both opportunity and challenge for the governance framework. LLM-generated SHACL shapes, if integrated through a human-in-the-loop review process, could accelerate constraint authoring and extend coverage to constraint types that are currently difficult to express

manually. D'Souza et al. [15] note that current LLMs lack formal correctness guarantees for generated constraint specifications -- a limitation significant in regulatory-grade governance contexts. The governance layer rule review workflow, which already requires steward sign-off on new rules before they enter the active catalogue, provides exactly the human-in-the-loop oversight mechanism needed to integrate LLM-assisted rule generation safely within an auditable governance framework. This extensibility was a deliberate architectural choice, anticipating the trajectory identified in the systematic review literature.

10. CONCLUSION

This paper has presented a complete, deployable framework for continuous validation and governance of enterprise ontologies through SPARQL-based automation pipelines. The four-layer reference architecture -- Ontology, Validation, Pipeline, and Governance -- provides a principled separation of concerns enabling independent evolution of technology components while maintaining coherent governance semantics across the full system. The hybrid SPARQL+SHACL validation engine achieves comprehensive constraint coverage across the six primary ontology quality dimensions identified in the literature [9], while the CI/CD integration shifts quality assurance from episodic pre-release review to continuous change-point enforcement. The governance layer operationalizes organizational quality policy through tiered response tiers, PROV-O provenance tracking, semantic versioning, and a real-time stewardship dashboard, establishing an auditable and accountable governance model appropriate for regulatory-grade vocabulary management.

Empirical evaluation against the NCI EVS production environment -- 170,247 concepts, 14.8 million triples, 300-500 weekly modifications -- demonstrated validation cycle time reductions of 58-62%, a 15.3-fold throughput improvement, and a 94.1% anomaly recall rate over a six-month operational deployment. The case study confirmed that continuous governance monitoring surfaces quality patterns -- chronic annotation completeness drift, systematic contributor workflow gaps -- that periodic review cannot reliably detect, demonstrating qualitative value beyond the quantitative performance improvements. The tiered policy engine successfully automated 94.3% of governance responses without steward intervention, reserving human judgment for genuinely ambiguous cases where expert context adds value beyond rule-based classification.

Future work will pursue three principal directions. First, integration of SWRL [10]-based OWL reasoning into the validation engine will extend constraint coverage to ontologies with complex axiom sets requiring inference rather than graph pattern matching. Second, LLM-assisted constraint generation [14, 15], mediated through the governance layer human-in-the-loop rule review workflow, will be evaluated for accelerating constraint authoring for new ontology domains. Third, extension of the governance framework to support cross-organizational ontology federation scenarios -- as envisioned by the OBO Foundry model [12] and emerging in enterprise knowledge graph federation contexts [13] -- will address the growing need for inter-institutional governance coordination in biomedical data integration. The framework architecture, rule catalogue, and operational lessons from the NCI EVS deployment will be made available to the biomedical ontology community as an open governance reference implementation.

REFERENCES

[1] S. Harris and A. Seaborne, Eds., "SPARQL 1.1 query language," W3C Recommendation, World Wide Web Consortium, Mar. 2013. [Online]. Available: https://www.w3.org/TR/sparql11-query/

[2] H. Knublauch and D. Kontokostas, Eds., "Shapes constraint language (SHACL)," W3C Recommendation, World Wide Web Consortium, Jul. 2017. [Online]. Available: https://www.w3.org/TR/shacl/

[3] R. Frosini et al., "Optimisation techniques for flexible SPARQL queries," ACM Trans. Web, vol. 16, no. 4, pp. 1-44, Nov. 2022. [Online]. Available: https://doi.org/10.1145/3532855

[4] M. Figuera et al., "Trav-SHACL: Efficiently validating networks of SHACL constraints," in Proc. Web Conf. 2021 (WWW 21), Ljubljana, Slovenia, Apr. 2021, pp. 3337-3348. [Online]. Available: https://doi.org/10.1145/3442381.3449877

[5] R. Meissner and K. Junghanns, "Using DevOps principles to continuously monitor RDF data quality," in Proc. 12th Int. Conf. Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, Sep. 2016, pp. 189-192. [Online]. Available: https://doi.org/10.1145/2993318.2993351

[6] M. Lefrancois and D. Gnabasik, "The SAREF pipeline and portal -- an ontology verification framework," in The Semantic Web - ISWC 2023, Lecture Notes in Computer Science, vol. 14266, Springer, Cham, 2023, pp. 134-151. [Online]. Available: https://doi.org/10.1007/978-3-031-47243-5_8

[7] N. Sioutos et al., "NCI Thesaurus: A semantic model integrating cancer-related clinical and molecular information," J. Biomed. Inform., vol. 40, no. 1, pp. 30-43, Feb. 2007. [Online]. Available: https://doi.org/10.1016/j.jbi.2006.02.013

[8] O. Erling and I. Mikhailov, "RDF support in the Virtuoso DBMS," in Networked Knowledge - Networked Media, Studies in Computational Intelligence, vol. 221, Springer, Berlin, Heidelberg, 2009, pp. 7-24. [Online]. Available: https://doi.org/10.1007/978-3-642-02184-8_2

[9] R. S. I. Wilson et al., "Analysis of ontology quality dimensions, criteria and metrics," in Computational Science and Its Applications - ICCSA 2021, Lecture Notes in Computer Science, vol. 12951, Springer, Cham, 2021, pp. 320-337. [Online]. Available: https://doi.org/10.1007/978-3-030-86970-0_23

I. Horrocks et al., "SWRL: A semantic web rule language combining OWL and RuleML," W3C Member Submission, World Wide Web Consortium, May 2004. [Online]. Available: https://www.w3.org/submissions/SWRL/

[10] [11] B. De Meester et al., "RDF graph validation using rule-based reasoning," Semantic Web, vol. 11, no. 6, pp. 1-28, 2020. [Online]. Available: https://doi.org/10.3233/SW-200384

[11] B. Smith et al., "The OBO Foundry: Coordinated evolution of ontologies to support biomedical data integration," Nat. Biotechnol., vol. 25, pp. 1251-1255, 2007. [Online]. Available: https://doi.org/10.1038/nbt1346

[12] B. Xue and L. Zou, "Knowledge graph quality management: A comprehensive survey," IEEE Trans. Knowl. Data Eng., vol. 35, no. 5, pp. 4969-4988, May 2023. [Online]. Available: https://doi.org/10.1109/TKDE.2022.3150080

[13] H. Babaei Giglou et al., "LLMs4OL: Large language models for ontology learning," in The Semantic Web - ISWC 2023, Lecture Notes in Computer Science, vol. 14265, Springer, Cham, 2023, pp. 408-427. [Online]. Available: https://doi.org/10.1007/978-3-031-47240-4_22

[14] J. D'Souza et al., "Large language models for ontology engineering: A systematic literature review," Semantic Web J., 2025 (in press). [Online]. Available: https://www.semantic-web-journal.net/content/large-language-models-ontology-engineering-systematic-literature-review