Provenance, Pragmatism, and Missing Axioms: A Critical Assessment of LLM-Mediated Knowledge Consolidation

A Commentary on McGreggor (2026)
Working Paper — Draft for Discussion
February 2026

Abstract

McGreggor (2026) presents a multi-stage pipeline for transforming primary source material into a structured, queryable knowledge base optimised for large language model consumption. The pipeline—developed through iterative experimentation rather than from theoretical first principles—produces concept cards, synthesised guides, a formal knowledge graph, and a unified search index, all served through a Model Context Protocol server. This commentary situates the pipeline within six converging research traditions: ontology engineering, knowledge engineering, library and information science, cognitive architecture, knowledge space theory, and the emerging field of LLM-mediated knowledge construction. We find that the pipeline independently rediscovers several well-established principles from these traditions while introducing genuinely novel elements at their intersection. We also identify significant methodological omissions—notably the absence of competency questions, formal ontology evaluation, confidence scoring, and authority control—that limit the system's robustness and scalability. We offer these observations not as grounds for dismissal but as a map of the theoretical terrain the pipeline inhabits, with the aim of strengthening future iterations.

1. Introduction

McGreggor's pipeline paper is refreshingly honest about its origins. The system was not derived from first principles; it was built to solve a practical problem—getting reliable, well-grounded responses from an LLM during interactive sessions—and the design decisions that shaped it were driven by what worked rather than by what theory prescribed. This pragmatic provenance is both the paper's greatest strength and its most significant vulnerability. The strength lies in demonstrated utility: the system produces a working knowledge base that has been applied to a real domain. The vulnerability lies in the possibility that design decisions which appear general may be entangled with properties specific to the domain, the LLM, or the use case.

The purpose of this commentary is to provide the theoretical context that the original paper explicitly defers. We situate each stage of the pipeline and each structural property it exhibits within the relevant research traditions, identifying where the pipeline aligns with established best practices, where it diverges, where it introduces genuinely novel contributions, and—most importantly for future development—where established techniques could address its current limitations.

We organise this assessment around three questions. First: what has the pipeline rediscovered? Second: what has it missed? Third: what has it invented? The answers to these questions collectively define the pipeline's position in the intellectual landscape and suggest a concrete path toward methodological maturity.

2. What the Pipeline Rediscovers

The pipeline's pragmatic development process has led it to converge, independently, on principles that have been established across multiple disciplines over the past six decades. This convergence is itself noteworthy: it suggests that these principles reflect genuine structural requirements of knowledge organisation rather than discipline-specific conventions.

2.1 Ontology Engineering: NeOn Scenario 5 and Methontology

The pipeline's overall architecture maps closely to **Scenario 5 of the NeOn Methodology** (Suárez-Figueroa, Gómez-Pérez, & Fernández-López, 2015): re-engineering non-ontological resources into structured knowledge representations. This scenario describes exactly what the pipeline does—transforming textbooks (non-ontological resources) into a structured knowledge base with formal properties. The stages of the pipeline also map, with recognisable correspondence, onto the lifecycle phases described in Fernández-López, Gómez-Pérez, and Juristo's Methontology (1997): source acquisition approximates the specification phase, concept extraction implements conceptualisation, and knowledge graph construction performs a form of informal formalisation.

The YAML-plus-Markdown dual representation implements, perhaps unwittingly, Gruber's principle of **minimal encoding bias** (Gruber, 1993)—the requirement that a knowledge representation should not privilege one encoding over another when multiple encodings could serve. By separating machine-readable metadata (YAML) from human-readable exposition (Markdown), the pipeline avoids forcing consumers into a single access pattern. This is a non-trivial design achievement, and the fact that it emerged from practical necessity rather than theoretical commitment makes it more, not less, credible as a design principle.

The provenance-preserving abstraction that McGreggor identifies as a structural property of the pipeline aligns with W3C's PROV-O ontology for provenance representation and with NeOn's provenance requirements. The chain from guide to concept card to source chapter to physical page implements what provenance researchers call a **derivation chain**—a sequence of transformations, each documented, that connects a derived artifact to its source.

2.2 Library and Information Science: Thesaurus Standards and Authority Control

From the perspective of library and information science, the concept cards are functionally **thesaurus term records** conforming, in spirit if not in formal compliance, to ANSI/NISO Z39.19. The internal sections of a concept card map to SKOS (Simple Knowledge Organization System) elements with recognisable directness: the quick definition corresponds to a preferred label with scope note; the formal definition to *skos:definition*; the examples section to *skos:example*; and the related concepts section to *skos:related*. The typed edges in the knowledge graph (prerequisite_of, extends, related) implement what cataloguers call **semantic relationships**—the broader-term, narrower-term, and related-term links that have been the structural backbone of controlled vocabularies since at least Roget.

The provenance chain from abstract concept through textbook formulation to physical PDF also implements, at a structural level, the Work-Expression-Manifestation-Item hierarchy described in IFLA's Functional Requirements for Bibliographic Records (FRBR). A concept such as "dominant seventh chord" exists as a Work; its treatment in a specific textbook constitutes an Expression; the PDF file is a Manifestation; a specific page reference identifies an Item. The pipeline does not use FRBR vocabulary, but it captures the same structural distinctions.

2.3 Cognitive Architecture: The Declarative–Procedural Distinction

The separation between concept cards (declarative) and guides (procedural) is perhaps the pipeline's most theoretically significant structural decision, and it maps precisely onto Anderson's ACT-R cognitive architecture (Anderson, 1983, 1993). In ACT-R, **declarative knowledge** consists of factual "chunks" stored in semantic networks—definitions, properties, relationships between entities. **Procedural knowledge** consists of production rules—condition-action pairs that enable skilled performance. The concept cards are declarative chunks; the guides are production-rule compositions.

This mapping is not merely analogical. ACT-R research demonstrates that procedural knowledge *compiles from* declarative knowledge through practice: learners first interpret declarative facts (a slow, effortful process), then gradually proceduralise them into automated performance patterns. The pipeline's guide synthesis process—composing multiple concept cards into integrated procedural guides—mirrors this compilation process. McGreggor does not claim this theoretical alignment, but it strengthens the case for the layered architecture as a design principle rather than a pragmatic accident.

Additional support comes from Merrill's Component Display Theory (1983) and Reigeluth's Elaboration Theory (1983), both of which prescribe instructional architectures that separate factual content from procedural integration. The pipeline's concept-card-to-guide progression implements what Reigeluth calls the movement from *epitome* (simplified overview) to *elaboration* (progressive detail) to *synthesis* (integrated understanding).

2.4 Knowledge Space Theory: Prerequisite Structures

The knowledge graph's prerequisite_of edges and the topological sort that computes learning orders place the pipeline in direct contact with Doignon and Falmagne's Knowledge Space Theory (KST), developed from the 1980s onward and formalised in their 1999 and 2011 monographs. KST defines a **knowledge space** as a domain of concepts Q together with a collection of subsets K (representing feasible knowledge states) satisfying specific closure properties. The prerequisite_of relation in the pipeline implements what KST calls a **surmise relation**—a partial ordering that determines which combinations of concepts are achievable.

The pipeline's topological sort generates what KST would call **learning paths**—sequences through the knowledge space that respect prerequisite constraints. Its centrality analysis identifies what KST terms **atoms**—the minimal knowledge states containing each concept. The practical system ALEKS, which implements KST for mathematics education, demonstrates the power of this framework: it models algebra as approximately 350 concepts generating millions of feasible states, with Markov-chain assessment identifying a student's current state in 25–30 questions.

The alignment is striking, but the pipeline does not verify the mathematical properties that KST requires. We return to this gap in Section 3.

3. What the Pipeline Misses

The pipeline's pragmatic origins mean that it was never subjected to the requirements analysis, evaluation protocols, and quality assurance processes that the relevant research traditions have developed over decades. The following omissions are not criticisms of the system's utility—which is demonstrated—but identifications of established practices whose absence limits robustness, scalability, and the ability to detect and correct errors systematically.

3.1 Competency Questions

The most consequential omission is the absence of **competency questions** (CQs)—the formal specification mechanism that Grüninger and Fox (1995) established as the foundational step in ontology development. Competency questions define what queries a knowledge system must be able to answer. They function simultaneously as requirements specification, scope delimitation, and acceptance criteria.

Without CQs, the pipeline extracts concepts from source material without a prior determination of what questions those concepts should answer. This creates two risks. First, **coverage gaps**: concepts that are needed to answer important queries may not be extracted because they are not prominent in the source material. Second, **scope creep**: concepts that are present in the source but irrelevant to the system's purpose may be extracted, increasing complexity without adding utility. Wisniewski et al. (2019) demonstrated that explicit competency questions significantly improve ontology coverage and reduce scope drift. Ren et al. (2014) showed that CQ-driven authoring improves consistency in ontology development.

For the music theory domain, illustrative competency questions might include: "What concepts must a student understand before learning sonata form?" "What are the common errors when applying voice-leading rules?" "How does the dominant seventh chord function differently in major and minor keys?" The pipeline should be evaluated against its ability to answer such questions, and gaps should drive extraction priorities.

3.2 Ontology Evaluation Frameworks

The pipeline currently has no mechanism for systematic quality assessment of its knowledge artifacts. Three established evaluation frameworks are relevant.

OntoClean (Guarino & Welty, 2002, 2004) evaluates taxonomic coherence by assigning **metaproperties** to concepts—rigidity (whether instances necessarily belong to the category), identity (what criteria distinguish instances), and unity (what holds instances together as wholes). These metaproperties constrain valid taxonomic relationships: a rigid concept cannot be subsumed under a non-rigid one, for example. Without such analysis, the pipeline's category taxonomy may contain structural inconsistencies that would be invisible to both human reviewers and the LLM consumer.

OOPS! (Poveda-Villalón, Gómez-Pérez, & Suárez-Figueroa, 2014) is an automated evaluation tool that scans ontologies for 41 catalogued pitfall patterns, including missing domain and range constraints, disconnected concepts, recursive definitions, and misused inverse relations. The pipeline's knowledge graph is susceptible to several of these pitfalls; automated scanning would catch errors that manual review misses.

Paulheim's refinement framework (2017) distinguishes two fundamental operations: *completion* (adding missing facts) and *error detection* (identifying incorrect facts). The pipeline performs neither operation explicitly. It extracts knowledge but does not systematically check for gaps (completion) or verify extracted assertions against multiple sources (error detection).

3.3 Relation Inventory Sparsity

The knowledge graph employs three edge types: prerequisite_of, extends, and related. This inventory is substantially sparser than what the represented knowledge likely requires. By comparison, SKOS defines seven core semantic relations; Wikidata employs over 10,000 properties; and even basic thesaurus standards (Z39.19) require at minimum broader-term, narrower-term, related-term, use, and used-for distinctions.

Relations that the domain almost certainly requires but that the current inventory cannot express include: **part_of / has_part** (a sonata form has an exposition, development, and recapitulation); **instance_of / exemplifies** (a specific Beethoven passage exemplifies sonata form); **same_as** for entity resolution across sources; **contrasts_with** for disambiguating commonly confused concepts; and **scaffolds** for relationships that are helpful but not strictly prerequisite. The "related" edge type currently functions as a catch-all for all non-prerequisite, non-extends relationships, collapsing semantically distinct connections into a single undifferentiated type.

3.4 Confidence Scoring

Major knowledge graph projects—YAGO, Knowledge Vault, NELL—assign confidence scores to extracted assertions, reflecting the reliability of the extraction process. The pipeline's LLM-mediated extraction produces no uncertainty signal. Every extracted concept card and every asserted relationship carries equal implicit confidence, regardless of how clear or ambiguous the source material was, how central or peripheral the concept is in the source, or how consistently the LLM handles similar extractions across runs.

Research on LLM extraction reliability has identified specific failure patterns. Tam et al. (2024) found 10–15% performance degradation when constraining LLMs to strict structured output during reasoning tasks. Huang et al. (2024) demonstrated that entity errors increase for rare entities and for facts mentioned later in generation sequences. Hallucination detection methods such as SelfCheckGPT provide mechanisms for identifying low-confidence extractions that should be flagged for human review. The absence of any such mechanism in the pipeline means that extraction errors propagate silently through all downstream stages.

3.5 Authority Control and Syndetic Structure

The pipeline lacks what library science calls **authority control**—the systematic management of variant forms of names and terms. In a music theory knowledge base, the same concept may be referred to as “dominant seventh chord,” “V7 chord,” “major-minor seventh,” or simply “V7.” Without explicit mapping between these variants, retrieval depends on the user employing the exact term used in the concept card. This is the problem that **syndetic structure**—the USE/UF (use/used-for) cross-reference system in controlled vocabularies—was designed to solve. Its absence creates predictable retrieval failures.

3.6 Knowledge Space Axiom Verification

As noted in Section 2.4, the pipeline’s prerequisite graph implements a structure that is close to—but does not formally satisfy—the requirements of Knowledge Space Theory. Specifically, the pipeline does not verify **closure under union**: the property that if knowledge states A and B are both feasible (i.e., learnable with their prerequisites), then the union $A \cup B$ must also be feasible. Without this verification, the prerequisite graph may contain configurations in which two valid learning paths, when combined, produce an impossible knowledge state. This would manifest as contradictory or incoherent prerequisite chains that are individually valid but jointly inconsistent.

3.7 Ontology Reuse and Alignment

NeOn Methodology identifies ontology reuse as a core scenario in most knowledge engineering projects—most domains have existing ontological resources worth incorporating rather than rebuilding from scratch. For music theory, relevant resources include the Music Ontology (Raimond et al.), JAMS (JSON Annotated Music Specification), and various vocabularies from the music information retrieval community. The pipeline builds entirely from primary sources without reference to existing structured representations, missing opportunities for alignment, interoperability, and validation against prior formalisation efforts.

4. What the Pipeline Invents

Not everything in the pipeline maps to existing practice. Several elements appear to be genuinely novel contributions, or at minimum, novel combinations of existing ideas applied in an underexplored context.

4.1 Unified Dual-Format Ontology Artifacts

The YAML-plus-Markdown concept card format has sparse precedent in the knowledge engineering literature. Ontology methodologies typically produce either formal artifacts (OWL files, RDF graphs) or documentation (natural language descriptions)—rarely both in a single, unified document. The concept card's dual structure—machine-readable frontmatter and human-readable body in one file—enables a workflow in which LLM extraction produces structured data and readable exposition simultaneously, without a separate documentation phase. This has practical affinities with Knuth's literate programming and with polyglot persistence patterns in software engineering, but its application to knowledge engineering artifacts is, to our knowledge, novel.

The pattern is particularly well-suited to LLM-mediated workflows. Tam et al.'s (2024) finding that strict JSON constraints degrade LLM reasoning performance suggests that a two-phase extraction pattern—free-form reasoning in natural language followed by structured formatting—would outperform purely structured extraction. The concept card's Markdown body provides a natural space for the reasoning phase, while the YAML frontmatter captures the structured output.

4.2 Procedural Guide Synthesis from Declarative Knowledge Bases

The automated or semi-automated synthesis of procedural guides from declarative concept cards is an underexplored operation. Ontology engineering focuses overwhelmingly on declarative structures: concepts, relations, axioms, taxonomies. The pipeline's guide synthesis—composing multiple concept cards into documents that describe how to use concepts together to accomplish practical goals—bridges declarative ontologies and procedural task support in ways that the existing literature rarely addresses directly. The closest analogues are CommonKADS's task model (Schreiber et al., 2000) and the worked example generation literature in intelligent tutoring systems, but neither describes LLM-mediated composition from a structured declarative base.

4.3 MCP as Knowledge Service Interface

The use of Model Context Protocol (Anthropic, 2024) as a serving layer for structured knowledge is too recent for substantial academic literature. The pipeline's implementation—providing graph algorithms, search, concept retrieval, and guide access through a standardised protocol that an LLM can invoke during interactive sessions—represents an emerging pattern in LLM-knowledge integration. Microsoft's GraphRAG (Edge et al., 2024) pursues similar goals—constructing knowledge graphs specifically for LLM retrieval augmentation—but MCP provides standardised, tool-level interfaces that GraphRAG's architecture lacks.

5. Assessment Summary

Figure 1 summarises the pipeline’s position relative to the relevant research traditions.

```
          PIPELINE ASSESSMENT MATRIX

REDISCOVERED              MISSED             INVENTED
(aligned with lit.)   (should add)       (novel contrib.)

• NeOn Scenario 5      • Competency Qs   • Dual-format
• Methontology          • OntoClean          cards
  lifecycle             • OOPS! pitfalls  • Procedural
• Gruber’s minimal      • Confidence         guide synth.
  encoding bias           scoring         • MCP as KG
• PROV-O provenance    • Authority ctrl     service API
• SKOS/thesaurus       • Richer relation
  structure               inventory
• FRBR hierarchy       • KST axiom
• ACT-R declarative/     verification
  procedural split     • Ontology reuse
• KST surmise          • Feedback loops
  relations
```

Figure 1. Assessment matrix showing the pipeline’s position relative to established research.

The overall assessment is encouraging. The pipeline has independently rediscovered a substantial number of established principles, which suggests that those principles reflect genuine structural requirements rather than disciplinary conventions. The omissions are addressable—none requires fundamental redesign—and the novel contributions, particularly the dual-format concept card and the procedural guide synthesis pattern, are worth formalising as contributions in their own right.

The pipeline’s most significant risk is not architectural but epistemological: the absence of systematic evaluation mechanisms means that errors introduced at any stage propagate silently through all downstream stages. Adding competency questions (to define requirements), confidence scoring (to flag uncertainty), and structural validation (to catch inconsistencies) would transform the pipeline from a working system into a *validated* working system—a distinction that matters considerably for both replication and extension to new domains.

6. Recommended Reading

For researchers or practitioners wishing to deepen their engagement with the theoretical traditions that bear on this pipeline, we recommend the following works, organised by the aspect of the pipeline they most directly address.

6.1 Ontology Methodology and Evaluation

Grüninger, M. & Fox, M.S. (1995). “Methodology for the Design and Evaluation of Ontologies.” IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing.

Suárez-Figueroa, M.C., Gómez-Pérez, A., & Fernández-López, M. (2015). “The NeOn Methodology Framework: A Scenario-Based Methodology for Ontology Development.” Applied Ontology, 10(2).

Guarino, N. & Welty, C. (2004). “An Overview of OntoClean.” In Handbook on Ontologies, Springer.

Poveda-Villalón, M., Gómez-Pérez, A., & Suárez-Figueroa, M.C. (2014). “OOPS! (OntOlogy Pitfall Scanner!).” IJSWIS, 10(2).

Wisniewski, D., Potoniec, J., Lawrynowicz, A., & Keet, C.M. (2019). “Analysis of Ontology Competency Questions and their Formalizations in SPARQL-OWL.” Journal of Web Semantics, 59.

6.2 Knowledge Engineering and Representation

Schreiber, G., et al. (2000). Knowledge Engineering and Management: The CommonKADS Methodology. MIT Press.

Gruber, T.R. (1993). “A Translation Approach to Portable Ontology Specifications.” Knowledge Acquisition, 5(2), 199–220.

Paulheim, H. (2017). “Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods.” Semantic Web, 8(3), 489–508.

6.3 Library and Information Science

Svenonius, E. (2000). The Intellectual Foundation of Information Organization. MIT Press.

Hjørland, B. (2002). “Domain Analysis in Information Science: Eleven Approaches.” JASIST, 53(6), 422–462.

Taylor, A.G. & Joudrey, D.N. (2009). The Organization of Information. 3rd ed. Libraries Unlimited.

6.4 Cognitive Architecture and Educational Theory

Anderson, J.R. (1993). Rules of the Mind. Lawrence Erlbaum Associates.

Falmagne, J.-C. & Doignon, J.-P. (2011). Learning Spaces: Interdisciplinary Applied Mathematics. Springer-Verlag.

Merrill, M.D. (1983). “Component Display Theory.” In Instructional Design Theories and Models, ed. C.M. Reigeluth.

6.5 LLM-Mediated Knowledge Engineering

Mungall, C., et al. (2024). “SPIRES: Structured Prompt Interrogation and Recursive Extraction of Semantics.” Bioinformatics.

Edge, D., et al. (2024). “From Local to Global: A Graph RAG Approach to Query-Focused Summarization.” Microsoft Research.

Pan, S., et al. (2024). “Unifying Large Language Models and Knowledge Graphs: A Roadmap.” IEEE TKDE.

Saeedizade, M. & Blomqvist, E. (2024). “Navigating Ontology Development with Large Language Models.” ESWC 2024.